Natural Version 3.1.4

Release Notes for Mainframes

SOFTWARE AG

Manual Order Number: NAT314-008IBB

This document applies to Natural Version 3.1.4 for Mainframes and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Readers’ comments are welcomed. Comments may be addressed to the Documentation Department at the address on the back cover or to the following e-mail address:

Documentation@softwareag.com

Table of Contents

Natural 3.1.4 Release Notes for Mainframes

⚠ For further information, you should also read the **Natural Version 2.3.3, Version 2.3.4, Version 3.1.2 and Version 3.1.3 Release Notes for Mainframes** which are delivered with Version 3.1.4 on the current Natural Documentation CD-ROM.
References contained in the Version 2.3 Release Notes generally refer to the corresponding Version 2.3 manuals which are not included in the Version 3.1.4 documentation set. Some additional links to the corresponding current version documents have also been provided where useful. Formal differences to the current release documentation may exist, for example, Software AG product names (Natural, Adabas, etc.) are written in capital letters.

This document covers the following topics:

- Introduction
- Prerequisites
- Migration Hints
- Documentation
- Maintenance End of Natural Versions
- Parameter Module
- Additional Values for Buffer Pool Text Size
- New Parameters in BS2STUB Macro
- TSENABL Option of the COMPOPT Command
- Natural for VSAM Installation Procedure
- Natural Security
- Shared Roll Files under IMS/TM
- Other Software AG Products
- Global Buffer Pool for VSE/ESA Environments
- Natural Optimizer Compiler
- Natural Turbo Performance Plug-In
- Review Natural Monitor
- Library SYSEXUEX
- Library SYSIDL

- Natural Web Interface for OS/390
- DRS Printing Facility
- German Error Messages
- Discontinued Delivery of Assembler Macros of Type E
- Information on Upcoming Releases

Introduction

These Release Notes inform you of the enhancements provided with Version 3.1.4 of Natural.

Natural 3.1.4 contains all ZAPs, INPL updates, early warnings and source changes applied to Natural 3.1.3 as error corrections.

Prerequisites

Operating/Teleprocessing Systems Required

Natural Version 3.1 requires the following versions of the following operating/teleprocessing systems:

Product	**Version**
BS2000/OSD	2 or above
OS/390	2.4 or above, for NaturalX: OS/390 Version 2.6 (or above) and OS/390 UNIX Services with access to the OS/390 HFS (hierarchical file system)
VSE/ESA	1.4.4 or above To use a global buffer pool: VSE/ESA Version 2.4 or above with the ESA/390 Subsystem Storage Protection Feature must be installed.
VM/ESA	2.1 or above
Com-plete	5.1.3 or above
CICS/ESA	4.1 or above It may be possible that Natural can be run with older CICS/ESA versions; however, Software AG cannot guarantee this.
CICS/VSE	2.3 or above
IMS/TM	3.1 or above
UTM	3.0 or above
TIAM	All versions available with OSD Version 2.

Assemblers Required

Natural Version 3.1 requires one of the following assemblers for the assembly of its source modules:

- "HL" assembler (IBM),
- "Assembh" assembler (Siemens).

It may well be possible that the source modules can be assembled with older assemblers; however, Software AG cannot guarantee this.

Migration Hints

When migrating from an earlier Natural version to Natural for Mainframes Version 3.1.4, you must observe the following hints:

Use of USR* programs

⚠ Make sure that the latest USR* programs from the delivered library SYSEXT are used.
If you have copied USR* programs to your FUSER system file, you have to replace them before use.

MAXBUFF Restrictions - RPC only

If you are using the ADALNK transport method to communicate with the EntireX Broker (default on mainframe), the following restrictions for MAXBUFF apply depending on the value specified for ACIVERS:

Value for ACIVERS	Maximum Value for MAXBUFF
ACIVERS=1	31
ACIVERS=2	30 (default value)
ACIVERS>2	30

For those migrating from Natural 2.3 to 3.1:

⚠ If MAXBUFF=31 has been used before you must adapt your NATPARM in one of the following two ways:

1. specify ACIVERS=1 and leave MAXBUFF untouched
2. specify MAXBUFF=30 and leave ACIVERS untouched

The latter is recommended to enable the support of EntireX Security.

Documentation

A completely updated and improved set of documents is distributed with this release. To remove redundancies, the documents for the following products or product components have been reorganized:

- NATUNLD/NATLOAD, SYSNCP
  These utilities are now described in documents that apply to all platforms.
- Entire Systems Server Interface
  The installation procedure published in the Natural for Mainframes Version 2.3.4 Release Notes is obsolete. See Installing the Entire System Server Interface (in the Natural for Mainframes Installation Guide).

In addition, the following Natural products have been provided with HTML-based documentation for online availability:

- Predict Version 4.1.2
- Super Natural Version 3.2.2
- Review Natural Monitor Version 3.6.1

These documents are available with the current edition of the Natural Documentation CD-ROM and will be published and updated via SAGnet as well.

Maintenance End of Natural Versions

- Software AG’s maintenance of Natural Versions 2.3.3, 2.3.4 and 3.1.2 for mainframes will end on 31 December 2000.
- Software AG’s maintenance of Natural Version 3.1.3 for mainframes will end on 30 June 2001.
- Software AG’s maintenance of Natural Optimizer Compiler Version 2.3.5 will end on 30 June 2001.

Software AG strongly recommends that you migrate to Natural Version 3.1.4 as soon as possible.

Parameter Module

NTDB Macro - Entire DB and Entire Transaction Propagator

If the same database is defined with an NTDB macro as Entire DB database (ENTIRE) and as Entire Transaction Propagator database (ETP), you must set the profile parameter ET=ON.

Additional Values for Buffer Pool Text Size

Additional values of 8, 12 and 16 KB have been introduced for the buffer pool text size.

Earlier Values:	Current Values:
1, 2, 4 (KB)	1, 2, 4, **8**, **12**, **16** (KB)

This applies to the following parameters:

NATBUFFER TSIZE parameter,
NTBPI macro TXSIZE parameter,
CMPSTART TXTSIZE parameter (BS2000/OSD).

New Parameters in BS2STUB Macro

The BS2STUB macro contains two new parameters, DESA and DATA to control allocation of a dataspace for the global buffer pool. Do not change their values unless you have specified the Natural parameter PLUGIN=BP.

TSENABL Option of the COMPOPT Command

The setting of the TSENABL option is currently ignored when a Natural object is executed. This will be corrected in the next major release of Natural.

Natural for VSAM Installation Procedure

If you are using VSAM system files, you must apply the special-purpose zaps NV44007 and NV44008 to the module NVSNUC before relinking all Natural nuclei.

Natural Security

Utility Profiles

Functional security for the SYSRPC utility has not yet been implemented for the mainframe platforms. It is available only with Natural for Windows, UNIX and OpenVMS.

Interface Subprograms for Retrieval

Interface subprograms for the display of library and user profiles are available, along with sample programs, from the menu of interface subprograms. These subprograms are NSCDA* and NSCDU.

Shared Roll Files under IMS/TM

If you do not use the roll server and you want to share roll files between Natural Version 2.3 and Version 3.1 environments, you have to format the roll files using the Natural Version 2.3 module NATRSFI. See Step 6 in section Installing the Natural IMS/TM Interface in the Natural Installation Guide for Mainframes.

Other Software AG Products

To use the following Software AG products in conjunction with Natural Version 3.1, the following product versions (or above) are required:

Product	Version
Adabas	6.2.1
Adabas Online Services	6.2.1
Adabas SQL Server	1.3.3
Adabas Stored Procedures And Triggers	6.2.2
Adabas Text Retrieval System	2.1.4
Com-plete	5.1.3 or above For Com-plete Version 5.1.3, the ZAPs CP51413 and CP51414 must be applied.
Con-form	3.2.3 or above
Con-nect	3.2.3 or above
Entire Broker Stub	5.1.1 For support of ACI Version 4, Entire Broker stub Version 5.2.1 is required.
Entire Event Management	2.1.2
Entire Network	5.3.1 with ZAP WX53052 and above (This product is required if you are using Natural Security in a heterogeneous environment.)
Entire Operations	3.1.1 Update 4
Entire Output Management	1.4.1 Update 1

Product	Version
Entire System Server	2.1.4
Entire System Server Interface	The Entire System Server Interface (ESX) is no longer a separate product but part of Natural (and included on the Natural installation tape). See also Installing the Entire System Server Interface in the Natural Installation Guide for Mainframes.
Entire Transaction Propagator	1.4.1
EntireX DCOM	5.3.1 (required for NaturalX only)
Natural Advanced Facilities	2.3.6 *
Natural CICS Interface	2.3.6 *
Natural Connection	3.1.4
Natural Construct	4.3.1
Natural Document Management	1.6.3
Natural for DB2	3.1.4
Natural for DL/I	2.3.6 *
Natural for SQL/DS	3.1.4
Natural for VSAM	2.4.6 *
Natural IMS/TM interface	2.3.6 *
Natural ISPF	2.4.2
Natural Optimizer Compiler	2.3.6

Product	Version
Natural Security for Mainframes	3.1.4
Natural TIAM interface	3.1.4
Natural TSO interface	3.1.4
Natural UTM interface	3.1.4
Natural VM/CMS Interface	3.1.4
NaturalX	3.1.4
Predict	3.4.2
Predict Application Control	2.1.4
Predict Case	2.5.2
Review Natural Monitor	3.6.1
Super Natural	3.2.1 or above

* Installation of this product version together with Natural Version 2.3.4 is not supported by Software AG.

Global Buffer Pool for VSE/ESA Environments

With Natural Version 3.1.4, VSE/ESA environments can take advantage of global buffer pools.

A global buffer pool in a VSE/ESA environment is a segment of storage assigned from the VSE/ESA system GETVIS area above or below 16 MB. A global buffer pool allows multiple Natural environments to share the same buffer pool to reduce storage requirements and system file access.

For more information, see Global Buffer Pool under VSE/ESA in the Natural Operations for Mainframes documentation.

Natural Optimizer Compiler

With Natural Version 3.1.4, Natural Optimizer Compiler Version 3.1.4 is available as a beta test release which includes the major enhancements described below.

The Natural Version 3.1.4 product comes with the Natural Optimizer Compiler Version 2.3.6, however, with the new Version 3.1.4 utilities being already integrated. The Natural Optimizer Compiler documentation supplied refers to Version 3.1.4, but is valid for Version 2.3.6, too. All features available with Version 3.1.4 only are marked correspondingly.

If you are interested in participating in the Natural Optimizer Compiler Beta Test, please contact force@softwareag.com.

Version 3.1.4 of the Natural Optimizer Compiler offers the following major enhancements:

- Program execution is speeded up significantly by means of caching variable values internally.
- The EXAMINE statement is optimized for certain clauses.
- Additional options are provided to give more control over how code is generated.
- The effect of the NOSGNTR option has been revised, also respecting the value of the PSIGNF compiler option.
- The NODBG option is also effective when INDEX, RANGE or OVFLW options are specified.
- The NOCSTAT command and the NOCSHOW utility have been improved.

In addition, miscellaneous other improvements increasing the degree of optimization have been made.

For more information on new features, enhancements and corrections, see the Natural Optimizer Compiler documentation.

Note:

- Natural objects cataloged with Natural Optimizer Compiler Version 2.3.x or with Natural Optimizer Compiler Version 2.1.6 (for Natural Version 2.2) can be executed under Natural Version 3.1.4.
- Natural objects cataloged with Natural Optimizer Compiler Version 3.1.4 can be executed under Natural Versions 3.1.2 and 3.1.3. However, Software AG strongly recommends that you migrate these environments to Natural Version 3.1.4 as soon as possible.

Natural Turbo Performance Plug-In

With Natural Version 3.1.4, an early version of a Natural turbo performance plug-in will be available for testing by selected customers. It is planned to make the plug-in generally available with Natural Version 3.1.5. The plug-in is available for the operating systems OS/390, VSE/ESA, BS2000/OSD and VM/ESA (except the buffer pool cache).

Significant performance improvements are provided by the following features:

- a new buffer pool search algorithm reduces the time required to seach for an object in the buffer pool,
- a buffer pool cache reduces system file accesses,
- using the Adabas multi-fetch option speeds up the process of loading objects into the buffer pool.

The performance plug-in must be explicitly enabled to become functional. For more information, see the Natural Turbo Performance Plug-In documentation.

Review Natural Monitor

With Natural Version 3.1.4, Review Natural Monitor Version 3.6.1 is released.

The following new features and enhancements are available:

- User interface like that of Natural Version 3.1.
- Runs stand-alone in library SYSRNM.

As a prerequisite, Review Natural Monitor Version 3.6.1 requires Natural for Mainframes Version 3.1.4.

For more information, see the Review Natural Monitor documentation.

Library SYSEXUEX

A new Natural library named SYSEXUEX, currently containing style sheets for NOCSTAT and for future add-ons, has been provided.

Library SYSIDL

With Natural Version 3.1.4, the INPL for the FNAT system file contains the library SYSIDL. This library is required only if EntireX is installed. It is provided for use by the EntireX Natural IDL Generator. Please refer to the EntireX documentation for more information.

Natural Web Interface for OS/390

The Natural Web Interface Server Extensions for CGI Interface have been ported to the OS/390 Unix System Services (USS). This eliminates the need to have a Unix- or Windows-based (HTTP) server to run the Natural Web Interface on the mainframe.

The Natural Web Interface Server Extensions are delivered with Natural Version 3.1.4 as CPIO file. The latest documentation updates are published in SAGnet: http://www.softwareag.com/sagnet/default.htm. Click to Recently Added Documentation and select the item from the selection box.

DRS Printing Facility

Software AG introduces Levi, Ray & Shoup, Inc. (LRS) as an Authorized Partner for Natural Printing Solutions. With the new release, Natural provides an interface to the LRS Enterprise Print Management family of products. This interface enables you to route the Natural print output to the LRS Dynamic Report System (DRS).

For more details, see also the LRS homepage at http://www.lrs.com/epm/drs.htm.

German Error Messages

The German Natural error messages and warnings were updated with the release of Natural Version 3.1.4.

Discontinued Delivery of Assembler Macros of Type E

With Natural CICS Interface Version 2.3.6, Natural for DL/I Version 2.3.6, Natural for VSAM Version 2.4.6 and Entire Review Natural Monitor Version 3.6.1, delivery of assembler macros of type E will be discontinued for VSE installations.

Information on Upcoming Releases

With the next major release of Natural (not with Version 3.1.5), it is planned to introduce the following:

- ENDING AT Clause in READ or HISTOGRAM Statements
- Discontinued Support of Applications Cataloged with Natural Version 2.2

ENDING AT Clause in READ or HISTOGRAM Statements

Internally, to determine the end of the range to be read, Natural reads one value beyond the ENDING AT value if an ENDING AT clause is specified for a READ or HISTOGRAM statement. This last record is in fact not the last record within the ENDING AT range, but the first record beyond that range (except if there is no further value after the ENDING AT value). THRU is a synonym of ENDING AT.

The next major release version of Natural will avoid reading beyond the specified end value. Only applications that access the last record read intentionally will be affected.

Example:

```
READ EMPLOYEES-VIEW BY NAME
     STARTING FROM 'MEIER'
     ENDING AT 'MEYER'
DISPLAY NAME
END-READ
DISPLAY EMPLOYEES-VIEW.NAME    /* will display 'MEYER' only if it is
                               /* the last record in the file
                               /* otherwise, the displayed value will be the
                               /* first value greater than 'MEYER'
```

To enable the user to choose between the old and the new behaviour, the keyword THRU will enforce that the loop is left with the last record within the specified value range, while ENDING AT will ensure that the value beyond the ENDING AT value is still read. This feature will apply only to Natural objects that are compiled with the new Natural version.

For compatibility reasons, a new option may be specified with the COMPOPT utility to enforce the old behaviour even with the THRU clause. As an alternative, you can change the THRU into an ENDING AT clause.

Discontinued Support of Applications Cataloged with Natural Version 2.2

With the next major release of Natural, applications that were cataloged with NATURAL Version 2.2 can be executed without any conversion procedure being required, and without your having to make any adjustments to the programs except for the ones mentioned above.

However, a future version of Natural may require that all such applications are recataloged to be executed with that version.

Software AG strongly recommends that existing applications be recataloged with Natural Version 3.1 to take advantage of improved runtime handling for such objects.

Natural Turbo Performance Plug-In

(Controlled Release Documentation)

Natural Turbo Performance Plug-In - Overview

With Natural Version 3.1.4, an early, controlled-release version of a Natural Turbo Performance Plug-In is available for testing by selected customers. It is planned to make the plug-in generally available with Natural Version 3.1.5. The plug-in is available for the operating systems OS/390, VSE/ESA, BS2000/OSD and VSE/ESA (except the buffer pool cache).

⚠ **Important Note:**
The Natural Turbo Performance Plug-In must be explicitly enabled to become functional. This is accomplished through the use of the new Natural parameter PLUGIN. In addition, since with the current version of Natural the turbo plug-in is available as a controlled release only, a specific licence key is required to activate it. Please, contact force@softwareag.com to obtain such a licence key.

This document contains special considerations that apply when you are using the Natural Performance Plug-In.

The following topics are covered:

- Introduction explains the new features that are available with the Natural Turbo Performance Plug-In and shows the results of the tests done by the Natural development team.
- Parameter PLUGIN describes the new Natural parameter PLUGIN that is required to enable the Natural Turbo Performance Plug-In.
- Natural Buffer Pool * provides information on the usage of the Natural buffer pool in conjunction with the Natural Turbo Performance Plug-In.
- Natural Global Buffer Pool under OS/390 * provides information on the usage of the Natural buffer pool in conjunction with the Natural Turbo Performance Plug-In.
- Natural Global Buffer Pool under VSE/ESA * provides information on the operation of a global buffer pool in conjunction with the Natural Turbo Performance Plug-In.
- Global Common Memory Pools under BS2000/OSD * describes the usage of the global common memory pools under BS2000/OSD when used in conjunction with the Natural Turbo Performance Plug-In.
- SYSBPMT Utility describes the new version of the SYSBPM utility.

* These sections are adapted excerpts from the Natural Operations documentation for mainframes.

Introduction

This document describes the new features that are introduced with the Natural Turbo Performance Plug-In and shows the results of the tests done by the Natural development team.

This document covers the following topics:

- Enhanced Buffer Pool Manager
- Buffer Pool Cache
- Enhanced Buffer Pool Management Utility SYSBPM
- Enhanced Program Loader
- Test Results

Enhanced Buffer Pool Manager

The enhanced buffer pool manager is activated by setting the Natural parameter PLUGIN to BP.

Objects residing in the buffer pool are located using a new search algorithm based on a hash table. This eliminates one of the most important performance bottlenecks, especially for environments using very large buffer pools (30 MB and more).

Rule of thumb: The more objects are loaded in the buffer pool, the higher the saving of CPU time using the enhanced buffer pool manager.

Buffer Pool Cache

The buffer pool cache is an optional feature. Objects residing in the buffer pool which have to be replaced by an other object due to missing space are cached in a data space. A subsequent load request for this object will be satisfied directly from the buffer pool cache. This reduces significantly the system file accesses, especially for rather small buffer pools or for sites using very large Natural objects.

> **Note:**
> The buffer pool cache is available in conjunction with a global buffer pool only. It is not available with VM/ESA.

Rule of thumb: If the size of the Natural global buffer pool is restricted due to any system requirements, expand the buffer pool by allocating a data space, which will be handled as a caching area.

Enhanced Buffer Pool Management Utility SYSBPM

The enhanced SYSBPM Utility "SYSBPMT" offers additional statistical information. In addition, it offers special statistic information about the new buffer pool cache.

Utility SYSBPM is still delivered, with Natural Version 3.1.5, however, it will be replaced by SYSBPMT under the name of SYSBPM. With the enhanced utility, it is possible to monitor buffer pools initialized by the enhanced buffer pool manager, however, additional statistical information made available by the enhanced buffer pool manager is not displayed. Therefore, it is recommended to use the new utility SYSBPMT in conjunction with the enhanced buffer pool manager, as buffer pools initialized by the standard buffer pool manager may be monitored as well by the new utility SYSBPMT.

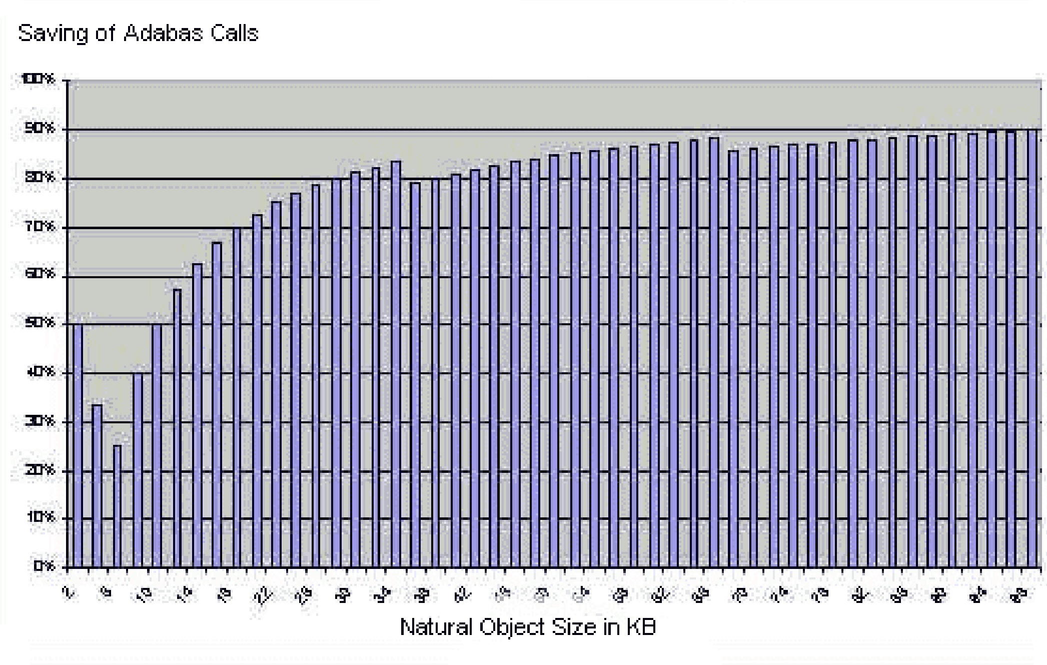
If a cache is used, the item "Total use" displayed in the "Object Directory Information" screen and the item "Total" displayed in the "Individual Object Statistics" screen are not lost if the object is removed from the buffer pool and saved to the cache. Therefore, these values indicate the number of uses of this object since it has been loaded from the system file. In addition, the load time displayed in the "Object Directory Information" screen is preserved, too, and constitutes the load time when the object was loaded from the systemfile. The "Total use" and "Total" counts can now be considerably higher than without cache.

Enhanced Program Loader

The enhanced program loader is activated by setting the Natural parameter PLUGIN to PL.

The new program loader uses the Adabas multi fetch option. This reduces significantly the number of Adabas calls necessary to load a Natural object, the other important performance bottleneck, especially in relation to the increased object size based on NAT2.3.

In the following chart, you will find the saving of Adabas calls with the enhanced program loader. In most of the installations, it will save more than 70% of Adabas calls to load the Natural objects into the Natural buffer pool.



The enhanced program loader and the buffer pool cache are of course somewhat contradictory, what means, that using a buffer pool cache reduces the number of program loads and therefore reduces the overall performance gain by the enhanced program loader itself.

Rule of thumb: The more objects that were loaded into the Natural buffer pool and the larger the objects are, the higher the saving of Adabas calls.

Test Results

Program Loader

The objective of these tests was to show the efficiency of the Natural Turbo Performance Plug-In program loader called enhanced program loader in the following. Therefore the chosen scenarios are theoretical and do not correspond to a real customer environment. The goal was to show the maximal possible performance gain of the enhanced program loader and to show where the enhanced program loader has no effect. All tests were executed using the standard buffer pool manager.

The basic environment common to both test scenarios:

- Two large objects, about 100 K, are invoked in a loop with 5000 iterations, i.e. we have 10.000 object executions. Loading an object of the chosen size requires 46 Adabas calls using the standard program loader and 6 Adabas calls using the enhanced program loader.
- All tests are done via execution of a batch job. A driver program is executed using the RUN command, therefore the executable of the driver program is NOT loaded into the buffer pool. The driver program then performs the CALLNATs to the given two objects.

The two different scenarios are:

Scenario 1

This is the ideal scenario for the enhanced program loader and shows the maximum possible performance increase by the enhanced program loader:

- A small buffer pool is used, i.e. every execution of the objects requires a reload from the Natural system file.

Scenario 2

This is the "worst" scenario for the enhanced program loader and demonstrates that in this case the enhanced program loader does not increase performance at all:

- A well sized buffer is used, i.e. the objects are loaded once and afterwards executed directly from the buffer pool.

	Scenario 1		**Scenario 2**	
	Standard loader	Enhanced loader	Standard loader	Enhanced loader
Number of database calls	460 000	60 000	92	12
Number of calls to the buffer pool manager to store the object	450 000	40 000	90	8
CPU time in seconds	0.74	0.31	0.01	0.01
Elapsed time in seconds	5.6	2.2	n.a	n.a
Time of the whole job in seconds	44.56	18.74	0.76	0.77

Some additional explanations**:**

The elapsed time indicated above is the elapsed time spent by the task control block used by the Natural task **without** the time spent waiting on receiving back control from Adabas.

The time of the whole job on the other hand is the total elapsed time used by the batch job **including** the time spent by cross-address space communication and by Adabas.

Buffer Pool Manager

The goal of these tests is to show the efficiency of a hash table by measuring especially the so-called locate calls, which means the time needed to detect if the requested object resides within the buffer pool.

The basic environment common to all five scenarios:

- 966 objects residing in the buffer pool, the buffer pool is full, no free text blocks are available.
- The indicated CPU time is the time spent in the buffer pool manager only.

Scenario 1 966.000 locates or 1000 locates for each object residing in the buffer pool

Scenario 2 100.000 locates for objects **not** residing in the buffer pool.

Scenario 3 965.000 locates for the 193 objects residing at the top of the buffer pool.

Scenario 4 965.000 locates for the 193 objects residing in the middle of the buffer pool

Scenario 5 965.000 locates for the 193 objects residing at the bottom of the buffer pool.

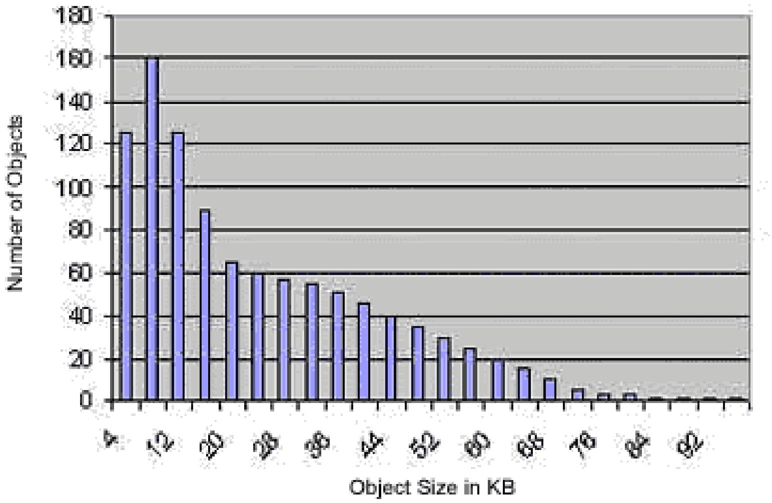
The table below shows the results of the enhanced versus the standard buffer pool manager (BPM):

	Enhanced BPM	**Standard BPM**
	CPU time in seconds	
Scenario 1	1.55	84.00
Scenario 2	0.11	8.56
Scenario 3	1.51	14.35
Scenario 4	1.57	76.49
Scenario 5	1.58	157.90

The results of the Scenarios 3 through 5 show very well that the time needed for locating an object is independent of the location of the object within the buffer pool.

Buffer Pool Cache Test

The test is based on a set of 1024 Natural programs of different sizes in the range from 4 KB to 96 KB. The number of programs in relation to the object size is shown in the following chart:



Each program calculates by the help of a random generator its successor and calls it using the FETCH statement. The random generator was always activated with the identical start value, therefore the sequence of program executions was the same in order to make comparable the different test runs with different parameter settings and different buffer pool and buffer pool cache sizes.

Sample 1

An object of the given set was executed 5000 times, which means the chain of programs invoking the next program randomly was iterated 5000 times. The program loader of the Natural Turbo Performance Plug-In was not used in Sample 1. If not expressly mentioned, a global buffer pool was used.

The results of Sample 1:

BP Type	Size (MB) BP	Size (MB) Cache	PLUGIN=	Time CPU	Time Elapsed	Finished Loads BP	Loads from Cache	Cache Get/Put Rate
Local	30		BP	3.43	30.81	1026		
	1	400	BP	3.81	24.94	1026	3605	3.51
	10	20	BP	4.34	24.90	1148	1465	1.28
	20		BP	5.04	60.03	1345		
	1	20	BP	4.86	59.28	1517	3114	2.05
	1	19	BP	4.67	38.26	1638	2997	1.83
	1	10	BP	6.94	65.46	2932	1705	0.58
	1	1	BP	9.54	143.72	4452	182	0.04
	1		BP	8.61	133.61	4612		
	1		NOBP	8.55	123.24	4632		

Some explanations:

Finished Loads BP The number of objects loaded from the Natural system file.

Loads from Cache The number of objects which could be retrieved from the buffer pool cache thus avoiding a program load from the Natural system file.

Cache Get/Put Rate The average number of times an object already residing in the buffer pool cache was reloaded into the buffer pool thus avoiding a subsequent reload from the Natural system file.

Sample 2

Same scenario as Sample 1, however with 20000 object executions.

BP Type	Size (MB) BP	Size (MB) Cache	PLUGIN=	Time CPU	Time Elapsed	Finished Loads BP	Loads from Cache	Cache Get/Put Rate
Local	30		BP	7.24	30.68	1031		
	1	400	BP	10.57	45.00	1031	17416	16.89
Local	30		NOBP	11.32	48.81	1031		

Sample 3

The test sequences of Sample 3 show the effect of the Natural Turbo Performance Plug-In program loader in conjunction with the buffer pool cache. As of course a large buffer pool in conjunction with a large buffer pool cache reduces the efficiency of the program loader, a small buffer pool and a small cache, if at all, was used.

In each case, a global buffer pool was used. 5000 objects were executed.

Size (MB) BP	Size (MB) Cache	PLUGIN=	Time CPU	Time Elapsed	Finished Loads BP
1	20	BP,PL	3.59	41.66	1493
1	20	BP,NOPL	4.47	31.21	1511
1		BP,PL	5.79	131.85	4628
1		BP,NOPL	9.00	158.87	4624

PLUGIN - Enable the Natural Plug-In Components

The PLUGIN parameter is required when you are using the Natural Turbo Performance Plug-In:

Possible Values	ON, OFF, PL, NOPL, BP, NOBP
Default Value	OFF

This parameter is used to enable the NATTURBO performance components. Currently there are 2 components:

PLUGIN=BP	The enhanced buffer pool manager, buffer pool cache and the enhanced SYSBPM application
PLUGIN=PL	The enhanced program loader

- The plug-in components can be activated or deactivated independently.
- The PLUGIN parameter can be specified as a single value or as a value list enclosed in brackets, e.g. PLUGIN=(BP,NOPL). The values NOBP and NOPL are intended to be used for dynamic specification to reset the PLUGIN setting of NATPARM. PLUGIN=ON activates all plug-in components and is equivalent to PLUGIN=(BP,PL). PLUGIN=OFF deactivates all plug-in components and is equivalent to PLUGIN=(NOBP,NOPL)
- Natural sessions using the default buffer pool manager can access only those buffer pools which were initialized with PLUGIN=NOBP.
- Natural sessions using the enhanced buffer pool manager can access any buffer pool.
- In multi-user environments using a local buffer pool (for example, under CICS), the PLUGIN setting of the first Natural session accessing this buffer pool determines whether the buffer pool is initialized with the default, i.e. the earlier version, (NOBP) or the enhanced buffer pool manager (BP); this means, with PLUGIN=BP, the buffer pool is initialized with hash table support.
- The enhanced program loader allocates a minimum size of 48K for the CFWSIZE internal buffer (the default value is 32K).

Natural Buffer Pool with Natural Turbo Performance Plug-In

This section provides information on Natural buffer pool management in conjunction with the Natural Turbo Performance Plug-In.

It covers the following topics:

- Hash Table
- Buffer Pool Cache

Hash Table

This section applies to buffer pools of TYPE=NAT only.

To speed up the search time for looking up an object in the buffer pool directory, an optional hash table is used. The number of entries in the hash table is twice the number of directory entries, rounded up to the next prime number. This will ensure that only half of the table is filled at any point in time and that the probability of collisions is near zero. As a consequence, the average number of tests to find an existing object in the hash table is theoretically less than 2.

The hash criterion is the eight character long program name. If more than one program name is hashed to the same location in the hash table, an overflow technique resolves the collisions.

When a programming object has to be loaded into the buffer pool, first the required space is allocated in the text pool. Afterwards, the name of the object is hashed to an entry in the hash table into which the pointer to the corresponding directory entry is saved.

When a programming object is searched for in the buffer pool, the program name of the object is hashed to an entry in the hash table and the corresponding directory entry is retrieved. If the directory entry contains the requested full object identifier (name, library, database ID and file number), the locate is successful. Otherwise, the locate has failed and the object has to be loaded into the buffer pool.

The storage required for the hash table is roughly 16 bytes per text block. Thus, the available storage in the text pool is reduced by between 1.6% (1 KB text blocks) and 0.1% (16 KB text blocks) compared to a buffer pool without hash table.

The usage of the hash table is optional and is available only in conjunction with the enhanced buffer pool manager. It can be turned on and off during buffer pool initialization with the new parameter PLUGIN.

- In the case of a global buffer pool, the PLUGIN setting specified with the startup parameters when starting the global buffer pool is used.
- In the case of a local buffer pool, the PLUGIN profile parameter setting of the first Natural session which connects to this buffer pool is used.
- In the case of the INITIALIZE command of the SYSBPM/SYBPMT utility, the PLUGIN setting of the current Natural session is used.

As a buffer pool with a hash table has an incompatible layout compared to a buffer pool without a hash table, only Natural sessions which use a compatible PLUGIN setting can access a buffer pool with a hash table.

The rules for the Natural sessions are summarized in the following chart:

PLUGIN=NOBP
PLUGIN=OFF
	The usage of the hash table is disabled and the buffer pool manager works in the same way as in the standard Natural version.
	If a Natural session tries to access a buffer pool initialized with PLUGIN=BP/ON, the access is rejected with error message 1071.
	If a Natural session has to initialize the buffer pool, this buffer pool is initialized without the hash table.

PLUGIN=BP
PLUGIN=ON
	The usage of the hash table is enabled.
	If a Natural session accesses a buffer pool initialized with PLUGIN=BP/ON, the fast access via the hash table is used.
	If a Natural session accesses a buffer pool initialized with PLUGIN=NOBP/OFF, this Natural session works in the same way as in the standard Natural version.
	If a Natural session has to initialize the buffer pool, this buffer pool is initialized with a hash table.

Problem Handling

In case of problems, the following fallback strategies using SYSBPMT are available:

- CLOSE HASH
  This command disables the hash table support and the enhanced buffer pool manager behaves in the same way as the default (earlier version) buffer pool manager. Nevertheless, it can be accessed only by sessions with PLUGIN=BP parameter setting.
- INITIALIZE OLD
  This command reinitializes the whole buffer pool without hash table support. This means, that each Natural session can access the buffer pool, regardless of its PLUGIN parameter setting.

In order to be always able to execute these commands, you are recommended to have an additional buffer pool initialized with PLUGIN=NOBP. This enables you to use SYSBPMT from a Natural session connected to this additional buffer pool and to issue commands to the affected buffer pool.

Buffer Pool Cache

The buffer pool cache option is only available in conjunction with a global buffer pool. It is used only for Natural programming objects (programs, subprograms, maps, etc.), whereas it is not used for example for objects generated by Natural for DL/I.

When a buffer pool is not large enough to take up all objects requested by the different users, special overload strategies are used to replace existing objects with requested objects. The number of overload situations has a direct relation to the efficiency of the buffer pool. The best and most efficient way to reduce the disliked overloads, hence to improve the buffer pool performance, is simply to increase its size.

However, this choice is not applicable at most customer sites, due to a lack of available storage in the primary address space and/or the MVS (E)CSA, VSE system GETVIS area or BS2000 Common Memory Pool.

In order to improve the situation described above, a buffer pool cache was introduced. The main purpose of this mechanism is to prevent a loss of all objects which were deleted from the buffer pool due to "short-on-buffer-pool-storage" situations. This means, that now an object delete results in a "swap out to buffer pool cache". The intended benefit of this feature is a reduction of database calls used for object loading and consequently a performance improvement.

The buffer pool cache area is allocated in a data space. When a data space is created for a buffer pool (BPCACHE parameter specified), the ownership is assigned to the creator task. If this task terminates, the system automatically deletes the data space. Therefore, the creator task will stay alive in this case, regardless of whether RESIDENT=Y has been set or not.

Natural Global Buffer Pool under OS/390

This section provides information on the Natural global buffer pool using the Natural Turbo Performance Plug-In.

The global buffer pool is a segment of storage assigned from the OS/390 extended common system area (ECSA) above 16 MB (or from CSA storage below, if requested), used by Natural to load and execute Natural programs.

Using a global buffer pool, multiple Naturals under different TP monitors (multiple copies of CICS, TSO, etc.) share the same area - thus requiring less storage than would be required for a local buffer pool in each environment.

This document covers the following topics:

- Operating the Natural Global Buffer Pool
- NATGBPMV Functions
- NATGBPMV Parameters
- Examples of NATBUFFER Specifications
- Sample NATGBPMV Execution Jobs
- Localization

Operating the Global Buffer Pool

The global buffer pool is operated by the program NATGBPMV which must be executed from an authorized library. NATGBPMV is contained in the Natural load library, with the authorization code already set to "1".

During the installation, the modules NATGBPMV and NATBPMGR are copied into an APF-authorized library.

⚠ Important Note:
To ensure that the global buffer pool is retained after a system failure, the global buffer pool should be started automatically during machine IPL.

The functions available from NATGBPMV are provided by a parameter card (PARM=), read from a file (see below) or entered by the MODIFY operator command unless NATGBPMV has not been terminated.

NATGBPMV expects the first command in the parameter field (PARM=) of the EXEC statement. You may enter one of the functions described below or a reference to an input file with "CF=<dd-name>", where <dd-name> represents a DD name defined in the JCL. Every line included from the input file represents a command. Blank lines or lines prefixed with an asterisk "*" are ignored. A file is processed until End-Of-File (EOF).

Example: PARM='CF=SYSIN1'

If the parameter field is not supplied or blank, the commands will be read from file SYSIN by default.

Each command received, either from the parameter card or using file input, is shown on the operator console.

After all commands are processed, the NATGBPMV terminates, unless

- RESIDENT=Y was specified or
- a buffer pool with a cache was created.

NATGBPMV Functions

The following functions are available:

CREATE | REFRESH | FSHUT | SHOWBP | TERMINATE

Note:
If no function is specified, CREATE is assumed.

CREATE - Create Global Buffer Pool

This function creates a global buffer pool with the specified parameters.

REFRESH - Re-initialize Global Buffer Pool

With the REFRESH command, it is possible to re-initialize an already active buffer pool. As no storage allocation takes place, the buffer pool size and location (above or below 16 MB) remain unchanged. However, it is possible to change the text block size (see NATBUFFER parameter).

FSHUT - Shut Down Global Buffer Pool

The global buffer pool is shut down, and the storage area is released.

If there are no active objects in the buffer pool, FSHUT is executed immediately.

If there are still active objects in the buffer, this will be indicated to the operator. Depending on the setting of the parameter CONFIRM, the operator is asked for a confirmation or FSHUT is executed immediately.

SHOWBP - Show Existing Buffer Pools

Displays all buffer pools currently existent.

TERMINATE - Terminate Program NATGBPMV

NATGBPMV is terminated. The termination of NATGBPMV does not affect any active global buffer pool.

NATGBPMV Parameters

The characteristics of a Natural global buffer pool can be defined with the aid of parameters. These parameters can be specified in any sequence. They can be abbreviated so that they are still unique.

> **Note:**
> If you like to start multiple global buffer pools with an associated cache, you are recommended to use a single starting task (NATGBPMV) and to supply the different CREATE commands by using an input dataset. See Example 4 below.

The following parameters are available:

BPNAME | BPLIST | BPCSIZE | CONFIRM | NATBUFFER | IDLE | PLUGIN | RESIDENT | SUBSID | TYPE

BPNAME - Name of Global Buffer Pool

This parameter is mandatory (except for the TERMINATE function). It specifies the name of the global buffer pool to be created.

BPNAME=*name*	*name* is the 8-byte name of the global buffer pool. If the specified name is shorter than 8 bytes, blanks will be appended to it.

BPLIST - Name of Preload List

This parameter specifies the name of the preload list.

BPLIST=*name*	*name* is the 8-byte name of the preload list. If the specified name is shorter than 8 bytes, blanks will be appended to it.

BPCSIZE - Buffer Pool Cache Data Space

This parameter specifies the amount of storage (in KB) used to allocate a data space for the buffer pool cache.

BPCSIZE=*size*	*size* is the amount of storage (in KB) used to allocate a data space for the buffer pool cache. The valid range is 100 - 2097148. If the BPCSIZE parameter is omitted (or set to zero), the buffer pool is not supplied with a cache. Note: A cache is only supported for buffer pools of type NAT.

CONFIRM - FSHUT Confirmation

This parameter controls the FSHUT behaviour if there are still active objects in the buffer pool.

CONFIRM=Y	A confirmation for the FSHUT function is required from the operator. The operator can decide to abort or to force the FSHUT function. This is the default value.
CONFIRM=N	FSHUT is forced without interaction with the operator.

NATBUFFER - Buffer Size, Mode, Text Block Size

This parameter specifies the size and the mode of the buffer pool, and the text block size.

NATBUFFER=*(size,mode,tsize)*	*size* is the amount of storage (in KB) to be allocated. The default is 100 KB; this is also the minimum possible size. *mode* determines if the global buffer pool is to be allocated above or below 16 MB. Possible values are: XA = above (default), BL = below. *tsize* determines the text block size (in KB). Possible values are: 1, 2, 4 ,8, 12, and 16. The default is 4. *size*, *mode* and *tsize* have to be specified in the sequence shown above.

If NATBUFFER is not specified, the default values will be used. See also Examples of NATBUFFER Specifications.

IDLE - Wait Time before Check

This parameter specifies the wait time before NATGBPMV checks if a further MODIFY command has been issued.

IDLE=*nn*	*nn* is the number of seconds to elapse before NATGBPMV checks if a further MODIFY command has been issued. The default is 10 seconds.

Once specified, IDLE will also apply to subsequent commands, without your having to specify it again. When the task owns buffer pool caches, NATGBPMV also checks for each buffer pool cache if its associated buffer pool is still active; if not that buffer pool cache is released; when the last buffer pool cache owned by the task has been released, the task terminates unless RESIDENT=Y has been specified.

PLUGIN - Enable Natural Turbo Performance Plug-In

This parameter must be used to enable the Natural Turbo Performance Plug-In.

PLUGIN=BP	The Natural Turbo Performance Plug-In is enabled. This means, the buffer pool is initialized with hash table support. **Attention:** You have to specify PLUGIN=BP in the Natural profile of each Natural session that is to access this buffer pool.
PLUGIN=NOBP	The Natural Turbo Performance Plug-In is not enabled. This means, the buffer pool is initialized without hash table support. This buffer pool can be accessed by any Natural session, regardless of its PLUGIN setting in its profile. This is the default setting.

RESIDENT - Behavior after Function Execution

This parameter specifies the behavior of NATGBPMV after the specified function has been executed. The following values are possible:

RESIDENT=Y	NATGBPMV will remain active after executing the specified function and await further commands. Once specified, RESIDENT=Y will also apply to subsequent commands, without your having to specify it again. (To stop NATGBPMV, you use the TERMINATE function.).
RESIDENT=N	NATGBPMV will terminate after executing the specified function, if no further command is available. If the task owns a buffer pool cache, RESIDENT=N is ignored and the task is not terminated.
RESIDENT=A	NATGBPMV automatically decides how to behave after having processed all commands. It will terminate if ● no further command is available and ● no buffer pool with an associated cache exists that was created by this task. In other words: If no buffer pool cache is owned by the task, RESIDENT=A works in the same way as RESIDENT=N. When the task owns a buffer pool cache, RESIDENT=A works the same way as RESIDENT=Y, but switches automatically to RESIDENT=N, when the last buffer pool whose associated buffer pool cache was owned by this task has terminated. This is the default setting.

SUBSID - Natural Subsystem ID

This parameter specifies the ID of the Natural subsystem.

SUBSID=*id*	*id* is the 4-byte ID of the Natural subsystem. Once specified, SUBSID will also apply to subsequent commands, without your having to specify it again. The default value is "NAT3".

If you use this parameter in conjunction with function SHOWBP, you may supply the value "*" to receive a buffer pool list for all existent subsystems.
For further information on the Natural subsystem, see Natural Subsystem.

TYPE - Type of Buffer Pool

This parameter specifies the type of the buffer pool. Possible values are:

TYPE=NAT	Natural buffer pool (this is the default).
TYPE=SORT	Sort buffer pool.
TYPE=DLI	DL/I buffer pool.
TYPE=EDIT	Editor buffer pool.
TYPE=MON	Monitor buffer pool.

Examples of NATBUFFER Specifications

To allocate a global buffer pool above 16 MB, with a size 1 MB and a text block size of 1 KB, you specify:
N=(1000,,1)

To allocate a global buffer pool above 16 MB, with a size 10 MB and a text block size of 4 KB, you specify:
N=(10000)

To allocate a global buffer pool above 16 MB, with a size 100 KB and a text block size of 4 KB, you specify:
N=(,,)

The third example is equivalent to omitting the NATBUFFER parameter altogether, as it causes the default values to apply.

NATGBPMV will return one of the following condition codes:

0	All functions executed successfully.
20	An error has occurred; see the message log for details.

Sample NATGBPMV Execution Jobs

The following examples show sample batch jobs for creating and terminating a global buffer pool.

Example 1:

```
//GBPSTART JOB
//*
//* Starts a global buffer pool with the name NAT31GBP, a size of 1 MB and
//* a text block size of 4 KB. The global buffer pool is allocated above 16 MB.
//* The subsystem used is NAT3.
//* After the allocation, the job GBPSTART terminates.
//*
//STEP EXEC PGM=NATGBPMV,PARM='BPN=NAT31GBP,N=(1000)'
//SETPLIB DD DISP=SHR,DNS=USER.APF.LINKLIB
```

Example 2:

```
//GBPRES JOB
//*
//* Starts a global buffer pool with the name 'GBP ', a default size of
//* 100 KB and a text block size of 1 KB. The global buffer pool is allocated
//* below 16 MB. The subsystem used is SAGS.
//* After the allocation, the job GBPRES will check every 30 seconds for
//* operator commands. Further commands may be entered using
//* the MODIFY operator command: F GBPRES,command-string
//*
//STEP EXEC PGM=NATGBPMV,PARM='BPN=GBP,N=(,BL,1),I=30,S=SAGS,R=Y'
```

Example 3:

```
//GBPSTOP
//*
//* Stops the global buffer pool GPB if it contains no active objects. If it
//* does contain active objects, the operator console will prompt for a reply.
//* Depending on the reply, the shutdown will be forced (Y) or aborted (N).
//* The subsystem used is NAT3.
//* the MODIFY operator command: F GBPRES,command-string
//*
//STEP EXEC PGM=NATGBPMV,PARM='FSHUT,BPN=GPB'
```

Example 4:

```
//GBPSTRT2
//* Read commands from SYSIN1:
//*
//* Start 3 global buffer pools (subsystem id N315) with name
//*   NATGBP1 - size=1024KB and a cache with size 2048KB
//*            (Initialized with implicit PLUGIN=BP due to specified BPC parm)
//*   NATGBP2 - size=2048KB with standard V31 layout
//*   NATGBP3 - size=1024KB without cache, but in new NatTurbo shape.
//* Display all buffer pools of subsystem id "N315".
//*
//* Note: The job does not terminate by itself, but stays resident and waits
//*       for operator commands, because it owns the data space allocated for
//*       buffer pool NATGBP1.
//*
//* If the buffer pools should shut down, send operator command MODIFY with
//* parameter "CF=SYSIN2" to execute the corresponding FSHUTs.
//*
//STEP EXEC PGM=NATGBPMV,PARM='CF=SYSIN1'
//SYSIN1  DD *
CREATE,BPN=NATGBP1,S=N315,N=(1024),BPC=2048
CREATE,BPN=NATGBP2,S=N315,N=(2048)
CREATE,BPN=NATGBP3,S=N315,N=(1024),PLUGIN=BP
SHOWBP S=N315
//SYSIN2 DD *
FSHUT,BPN=NATGBP1,S=N315
FSHUT,BPN=NATGBP2,S=N315
FSHUT BPN=NATGBP3,S=N315
SHOWBP S=N315
//*
```

Localization

The module NATGBPTX (already included in NATGBPMV) is delivered in source form. It contains all error messages in English. The messages can be translated into other languages as required; in this case, NATGBPMV has to be newly link-edited by replacing the CSECT NATGPBTX.

Natural Global Buffer Pool under VSE/ESA

This section provides information on the Natural global buffer pool using the Natural Turbo Performance Plug-In.

⚠ Important Note:
A Natural global buffer pool under VSE/ESA requires the subsystem storage protection facility of an ESA/390 or compatible processor. Consequently, it also requires a minimum operating system level of VSE/ESA Version 2 Release 4 for support of this hardware feature.

The global buffer pool is a segment of storage assigned from the VSE/ESA system GETVIS storage above 16 MB (or from storage below, if requested), used by Natural to load and execute Natural programs.

Using a global buffer pool, multiple Naturals under different TP monitors (multiple copies of CICS, Com-plete, etc.) share the same area - thus requiring less storage than would be required for a local buffer pool in each environment.

This document covers the following topics:

- Operating the Natural Global Buffer Pool
- NATGBPMG Functions
- NATGBPMG Parameters
- Examples of NATBUFFER Specifications
- Sample NATGBPMG Execution Jobs
- Localization

Operating the Global Buffer Pool

The global buffer pool is operated by program NATGBPMG which is executed from the Natural load library.

⚠ **Attention:**
To ensure that the global buffer pool is retained after a system failure, the global buffer pool should be started automatically during machine IPL.

The functions available from NATGBPMG are provided by a parameter card (PARM=), read from a file (see below) or supplied by the operator (AR command MSG xx with xx being the VSE partition ID) unless NATGBPMG has not been terminated.

NATGBPMG expects the first command in the parameter field (PARM=) of the EXEC statement. You may enter one of the NATGBPMG Functions described below or a reference to an input file with "CF=<dlbl-name>", where <dlbl-name> represents a DLBL name defined in the JCL or the VSE/ESA (partition) standard labels. Every line included from the input file represents a command. Blank lines or lines prefixed with an asterisk "*" are ignored. An asterisk ("*") for <dlbl-name> indicates to NATGBPMG that input has to be read from SYSIPT. A file is processed until End-Of-File (EOF).

Example: PARM=’CF=SYSIN1’

If the parameter field is not supplied or is blank, the commands will be read from SYSIPT by default.

Each command received from the parameter card, from file input or from operator console input is shown on the operator console and is logged to SYSLST.

After all commands are processed, the NATGBPMG terminates unless

- RESIDENT=Y was specified or
- a buffer pool with a cache was created.

NATGBPMG Functions

The following functions are available:

CREATE | REFRESH | FSHUT | SHOWBP | TERMINATE

Note:
If no function is specified, CREATE is assumed.

CREATE - Create Global Buffer Pool

This function creates a global buffer pool with the specified parameters.

REFRESH - Re-initialize Global Buffer Pool

With the REFRESH command it is possible to re-initialize an already active buffer pool. As no storage allocation takes place, the buffer pool size and location (above or below 16 MB) remain unchanged. However, it is possible to change the text-block size (see NATBUFFER parameter).

FSHUT - Shut Down Global Buffer Pool

The global buffer pool is shut down, and the storage area is released.

If there are no active objects in the buffer pool, FSHUT is executed immediately.

If there are still active objects in the buffer, this will be indicated to the operator. Depending on the setting of the parameter CONFIRM, the operator is asked for a confirmation or FSHUT is executed immediately.

SHOWBP - Show Existing Buffer Pools

Displays all buffer pools currently existing.

TERMINATE - Terminate Program NATGBPMG

NATGBPMG is terminated. The termination of NATGBPMG does **not** affect any active global buffer pool.

NATBPGMG Parameters

The characteristics of a Natural global buffer pool can be defined with the aid of parameters. These parameters can be specified in any sequence. They can be abbreviated so that they are still unique.

> Note:
> If you like to start multiple global buffer pools with an associated cache, you are recommended to use a single starting task (NATGBPMG) and to supply the different CREATE commands in an input dataset. See Example 4 below.

The following parameters are available:

BPNAME | BPLIST | BPCSIZE | CONFIRM | NATBUFFER | IDLE | PLUGIN | RESIDENT | SUBSID | TYPE

BPNAME - Name of Global Buffer Pool

This parameter is mandatory (except for the TERMINATE function). It specifies the name of the global buffer pool to be created.

BPNAME=*name*	*name* is the 8-byte name of the global buffer pool. If the specified name is shorter than 8 bytes, blanks will be appended to it.

BPLIST - Name of Preload List

This parameter specifies the name of the preload list.

BPLIST=*name*	*name* is the 8-byte name of the preload list. If the specified name is shorter than 8 bytes, blanks will be appended to it.

BPCSIZE - Buffer Pool Cache Data Space

This parameter specifies the amount of storage (in KB) used to allocate a data space for the buffer pool cache.

BPCSIZE=*size*	*size* is the amount of storage (in KB) used to allocate a data space for the buffer pool cache. The valid range is 100 - 2097148. If the BPCSIZE parameter is omitted (or set to zero), the buffer pool is not supplied with a cache. Note: A cache is only supported for buffer pools of TYPE=NAT.

CONFIRM - FSHUT Confirmation

This parameter controls the FSHUT behaviour if there are still active objects in the buffer pool.

CONFIRM=Y	A confirmation for the FSHUT function is required from the operator. The operator can decide to abort or to force the FSHUT function. This is the default value.
CONFIRM=N	FSHUT is forced without interaction with the operator.

NATBUFFER - Buffer Size, Mode, Text Block Size

This parameter specifies the size and the mode of the buffer pool, and the text block size.

NATBUFFER=*(size,mode,tsize)*	*size* is the amount of storage (in KB) to be allocated. The default is 100 KB; this is also the minimum possible size. *mode* determines if the global buffer pool is to be allocated above or below 16 MB. Possible values are: XA = above (default), BL = below. *tsize* determines the text block size (in KB). Possible values are: 1, 2, 4 ,8, 12, and 16. The default is 4. *size*, *mode* and *tsize* have to be specified in the sequence shown above.

If NATBUFFER is not specified, the default values will be used. See also Examples of NATBUFFER Specifications.

IDLE - Wait Time before Check

This parameter is ignored when the task does not own a buffer pool cache.

IDLE=*nn*	*nn* is the number of seconds to elapse before NATGBPMG checks for each buffer pool cache if its associated buffer pool is still active; if not that buffer pool cache is released; when the last buffer pool cache owned by the task has been released, the task terminates, unless RESIDENT=Y has been specified. The default setting is 10 seconds.

Once specified, IDLE will also apply to subsequent commands, without your having to specify it again.

PLUGIN - Enable Natural Turbo Performance Plug-In

This parameter must be used to enable the Natural Turbo Performance Plug-In.

PLUGIN=BP	The Natural Turbo Performance Plug-In is enabled. This means, the buffer pool is initialized with hash table support. Attention: You have to specify PLUGIN=BP in the Natural profile of each Natural session that is to access this buffer pool.
PLUGIN=NOBP	The Natural Turbo Performance Plug-In is not enabled. This means, the buffer pool is initialized without hash table support. This buffer pool can be accessed by any Natural session, regardless of its PLUGIN setting in its profile. This is the default setting.

RESIDENT - Behavior after Function Execution

This parameter specifies the behavior of NATGBPMV after the specified function has been executed. The following values are possible:

RESIDENT=Y	NATGBPMG will remain active after executing the specified function and await further commands. Once specified, RESIDENT=Y will also apply to subsequent commands, without your having to specify it again. (To stop NATGBPMG, you use the TERMINATE function.).
RESIDENT=N	NATGBPMG will terminate after executing the specified function, if no further command is available. If the task owns a buffer pool cache, RESIDENT=N is ignored and the task is not terminated.
RESIDENT=A	NATGBPMG automatically decides how to behave after having processed all commands. It will terminate if ● no further command is available and ● no buffer pool with an associated cache exists that was created by this task. In other words: If no buffer pool cache is owned by the task, RESIDENT=A works in the same way as RESIDENT=N. When the task owns a buffer pool cache, RESIDENT=A works the same way as RESIDENT=Y, but switches automatically to RESIDENT=N, when the last buffer pool whose associated buffer pool cache was owned by this task has terminated. This is the default setting.

SUBSID - Natural Subsystem ID

This parameter specifies the ID of the Natural subsystem.

SUBSID=*id*	*id* is the 4-byte ID of the Natural subsystem. Once specified, SUBSID will also apply to subsequent commands, without your having to specify it again. The default value is "NAT3".

If you use this parameter in conjunction with the function SHOWBP, you may supply the value "*" to receive a buffer pool list for all existent subsystems.
For further information on the Natural subsystem, see Natural Subsystem.

TYPE - Type of Buffer Pool

This parameter specifies the type of the buffer pool. Possible values are:

TYPE=NAT	Natural buffer pool (this is the default).
TYPE=SORT	Sort buffer pool.
TYPE=DLI	DL/I buffer pool.
TYPE=EDIT	Editor buffer pool.
TYPE=MON	Monitor buffer pool.

Examples of NATBUFFER Specifications

- To allocate a global buffer pool above 16 MB, with a size 1 MB and a text block size of 1 KB, you specify:
  `N=(1000,,1)`
- To allocate a global buffer pool above 16 MB, with a size 10 MB and a text block size of 4 KB, you specify:
  `N=(10000)`
- To allocate a global buffer pool above 16 MB, with a size 100 KB and a text block size of 4 KB, you specify:
  `N=(,,)`

The third example is equivalent to omitting the NATBUFFER parameter altogether, as it causes the default values to apply.

NATGBPMG will return one of the following condition codes:

0	All functions executed successfully.
20	An error has occurred; see the message log for details.

Sample NATGBPMG Execution Jobs

The following examples show sample batch jobs for creating and terminating a global buffer pool.

Example 1:

```
// JOB GBPSTART
/*
/* Starts a global buffer pool with the name NAT31GBP, a size of 1 MB and
/* a text block size of 4 KB. The global buffer pool is allocated above 16 MB.
/* The subsystem used is NAT3.
/* After the allocation, the job GBPSTART terminates.
/*
// LIBDEF PHASE,SEARCH=SAGLIB.NATLIB
// EXEC NATGBPMG,SIZE=NATGBPMG,PARM='BPN=NAT31GBP,N=(1000)'

/*

// EXEC LISTLOG

/&
```

Example 2:

```
// JOB GBPRES
/*
/* Starts a global buffer pool with the name ’GBP’, a default size of
/* 100 KB and a text block size of 1 KB. The global buffer pool is allocated
/* below 16 MB. The subsystem used is SAGS.
/* After the allocation, the job GBPRES will fall asleep.
/* Further commands may be entered using AR command MSG partition-id:
/* the job GBPRES will then prompt for console input.
/*

// LIBDEF PHASE,SEARCH=SAGLIB.NATLIB
// EXEC NATGBPMG,SIZE=NATGBPMG,PARM=’BPN=GBP,N=(,BL,1),S=SAGS,R=Y’
/*

// EXEC LISTLOG

/&
```

Example 3:

```
// JOB GBPSTOP
/*
/* Stops the global buffer pool GPB if it contains no active objects. If it
/* does contain active objects, the operator console will prompt for a reply.
/* Depending on the reply, the shutdown will be forced (Y) or aborted (N).
/* The subsystem used is NAT3.

/*
// LIBDEF PHASE,SEARCH=SAGLIB.NATLIB
// EXEC NATGBPMG,SIZE=NATGBPMG,PARM=’FSHUT,BPN=GBP’
/*

// EXEC LISTLOG

/&
```

Example 4:

```
// JOB GBPSTRT2
/* Read commands from SYSIPT:
/*
/* Start 3 global buffer pools (subsystem id N315) with name
/*   NATGBP1 - size=1024KB and a cache with size 2048KB
/*            (Initialized with implicit PLUGIN=BP due to specified BPC parm)
/*   NATGBP2 - size=2048KB with standard V31 layout
/*   NATGBP3 - size=1024KB without cache, but in new NatTurbo shape.
/* Display all buffer pools of subsystem id "N315".
/*
/* Note: The job does not terminate by itself, but stays resident and waits
/*       for operator commands, because it owns the data space allocated for
/*       buffer pool NATGBP1.
/*
/* If the buffer pools should shut down, wake up sleeping job by MSG partition-id
/* and enter parameter "CF=*" to execute the corresponding FSHUTs.
/*
// LIBDEF PHASE,SEARCH=SAGLIB.NATLIB
// EXEC NATGBPMG,SIZE=NATGBPMG

CREATE,BPN=NATGBP1,S=N315,N=(1024),BPC=2048
CREATE,BPN=NATGBP2,S=N315,N=(2048)
CREATE,BPN=NATGBP3,S=N315,N=(1024),PLUGIN=BP
SHOWBP S=N315
/*
FSHUT,BPN=NATGBP1,S=N315
FSHUT,BPN=NATGBP2,S=N315
FSHUT BPN=NATGBP3,S=N315
SHOWBP S=N315
/*
```

Localization

The module NATGBPTX (already included in NATGBPMG) is delivered in source form. It contains all error messages in English. The messages can be translated into other languages as required; in this case, NATGBPTX has to linked into the Natural Load library with phase name NATGBPTX..

Global Common Memory Pools under BS2000/OSD

⚠ This section contains special considerations that apply to the Global Common Memory Pools when running Natural under BS2000/OSD **and when using the Natural Turbo Performance Plug-In**.

The following topics are covered:

- New Keyword Parameters for CMPSTART and BS2STUB
- Rules for Using the New Keyword Parameters
- Global Common Memory Pools
- Local Common Memory Pools

New Keyword Parameters for CMPSTART and BS2STUB

The following parameters are applicable only if Natural has been set up for use with the Natural Turbo Performance Plug-In, i.e., when PLUG-IN=BP has been specified in the Natural parameter module.

To generate a global Natural buffer pool the following keyword parameters are used:

DESA - ESA Data Space Area

DESA=YES	An ESA data space area is to be created for the Natural buffer pool.
DESA=NO	No ESA data space area is to be created (this is the default value).

DATA - Size of Data Space Area

DATA=*nnn*MB	Specifies the size of the data space area in Megabytes.
DATA=*nnn*KB	Specifies the size of the data space area in Kilobytes.

By default, no size is specified.

PLUGIN - Initialize Hash Support

PLUGIN=BP	The Natural buffer pool is to be initialized for the new hash support of the buffer pool manager.
PLUGIN=NOBP	Hash support is not enabled (this is the default value).

Rules for Using the New Keyword Parameters

The parameter PLUGIN is valid only for the Natural parameter module and for the program CMPSTART.

When DESA=YES,DATA=*nnn*MB or DATA=*nnn*KB is specified for the start of a **global** buffer pool (CMPSTART), this implies the definition of PLUGIN=BP for the program CMPSTART.

To connect a Natural **global** buffer pool and the associated ESA data space, you need only set DESA=YES in the macro DADDON (for the generation of the module BS2STUB). The specification DATA=*nnn*MB or DATA=*nnn*KB is omitted.

The generation of an ESA data space for a Natural **local** buffer pool is admissible only for single-task operation, for example, in batch mode. In this case, to define a **local** buffer pool, you must specify the following in the macro ADDON:

STAT=LOCAL,SCOP=LOCAL,DESA=YES,DATA=nnnMB/KB

The Natural parameter module must be created with the setting PLUGIN=BP.

Example for Creating a Natural Global Buffer Pool with ESA Data Space

```
/.BP314GA LOGON
/OPTION DUMP=YES
/SYSFILE SYSOUT=LST.BP314GA
/SYSFILE SYSDTA=(SYSCMD)
/EXEC (CMPSTART,NAT314.BS2.MOD)
NAME=BP314GA,TYPE=NAT,POSI=ABOVE,SIZE=10MB,ADDR=260,DESA=YES
DATA=32MB,PLUGIN=BP
/SYSFILE SYSDATA=(PRIMARY)
/LOGOFF N
```

Example for Connecting a Natural Global Buffer Pool with ESA Data Space

```
STUB314A BS2STUB PARMOD=31,PROGMOD=ANY
         ADDON NAME=BP314GA,TYPE=NAT,STAT=GLOBAL,DESA=YES
```

Example for Creating and Connecting a Natural Local Buffer Pool with ESA Data Space

```
STUB314B BS2STUB PARMOD=31,PROGMOD=ANY
         ADDON NAME=BP314LA,TYPE=NAT,POSI=ABOVE,SIZE=10MB
               STAT=LOCAL,SCOP=LOCAL,DESA=YES,DATA=32MB
```

Global Common Memory Pools

The following programs are provided to start and stop global common memory pools in Natural under BS2000/OSD:

- CMPSTART
- CMPEND

CMPSTART Program

Program CMPSTART does the following:

- It starts global common memory pools with its own start task.
- It loads a defined module into a global common memory pool.
- It initializes a global common memory pool.

The keyword parameters TXTSIZE and BPLIST (see below) are only valid for program CMPSTART when starting a global Natural buffer pool.

All other keyword parameters are identical with the keyword parameters for the macro ADDON used for generating the module BS2STUB.

TXTSIZE - Buffer-Pool Text-Record Size

Possible values:	1, 2, 4, 8, 12, 16
Default value:	4

This parameter defines the Natural buffer-pool text-record size in KB.

BPLIST - Preload List For Global Buffer Pool

Possible values:	*name*
Default value:	none

This parameter defines the name of a preload list for a global Natural buffer pool. The defined Natural programs of the preload list will be loaded into the global Natural buffer pool when the first user logs on (see the Natural profile parameter BPLIST).

Operator Commands

These operator commands terminate a global common memory pool:

```
/INTR tsn,STOP
or:
/INTR tsn,END
```

This operator command displays the global common memory pool’s name, position, address, size and activation time on the console:

```
/INTR tsn,DPRM
```

This operator command terminates the global common memory pool’s start task with a dump:

```
/INTR tsn,DUMP
```

Examples:

To start a global load pool (shared nucleus):

```
/.NATSHRE LOGON
/OPTION DUMP=YES
/SYSFILE SYSDTA=(SYSCMD)
/SYSFILE SYSOUT=LST.NATSHARE
/EXEC (CMPSTART,NAT230.MOD)
NAME=NATSHARE,SIZE=2MB,POSI=ABOVE,ADDR=250,SCOP=GLOBAL
PFIX=YES,ALNK=NO,LIBR=NAT230.USER.MOD
/SYSFILE SYSDTA=(PRIMARY)
/LOGOFF
/* NATSHARE IS THE NAME OF THE LINKED NATURAL REENTRANT MODULE.
   IT IS ALSO THE
/* NAME OF THE COMMON MEMORY POOL. THE ADDRESS OF THE GLOBAL
   NATURAL LOAD POOL
/* MUST BE DEFINED. THE ADDRESS MUST BE FIXED (PFIX=YES).
```

To start a global Natural buffer pool:

```
/.BP230GA LOGON
/OPTION DUMP=YES
/SYSFILE SYSDTA=(SYSCMD)
/SYSFILE SYSOUT=LST.BP230GA
/EXEC (CMPSTART,NAT230.MOD)
NAME=BP230GA,TYPE=NAT,POSI=ABOVE,SIZE=2048KB,SCOP=GLOBAL
/SYSFILE SYSDTA=(PRIMARY)
/LOGOFF
/* FOR A NATURAL BUFFER POOL, THE OPERAND OF PARAMETER "TYPE" MUST
   BE DEFINED
/* AS 'NAT'.
```

CMPEND Program

Program CMPEND terminates the start tasks for all global common memory pools. The input for CMPEND are the names of the global common memory pools.

Example:

```
/SYSFILE SYSDTA=(SYSCMD)
/EXEC (CMPEND,NAT230.MOD)
NATSHARE,BP230GA
/* THE DELIMITER FOR THE DEFINED NAMES IS ’ ’ OR ’,’.
```

Local Common Memory Pools

The following macros enable you to define local (or global) common memory pools in Natural under BS2000/OSD:

- BS2STUB
- ADDON
- ADDEND

BS2STUB Macro

The macro BS2STUB does the following:

- Starts local common memory pools.
- Connects to a defined global common memory pool.
- Loads a defined module into a local common memory pool.
- Loads dynamically called 3GL programs.

BS2STUB has the following parameters:

```
name BS2STUB PARMOD=nn,PROGMOD=xx
```

name	Specifies the CSECT name. Default value: BS2STUB. The first three characters must not contain the value "NAT".
PARMOD	Specifies the addressing mode. Possible values: 24 or 31 (bit). Default value: 24.
PROGMOD	Specifies whether dynamically loaded programs are to be loaded above or below the 16-MB line. Possible values: ANY or 24. ANY means that the module is loaded above or below the 16-MB line. 24 means that the module is loaded below the 16-MB line.

ADDON Macro

The macro ADDON defines a common memory pool in the ADDON table of program BS2STUB. It contains the following keyword parameters which are also applicable to program CMPSTART:

ACCS | ADDR | ALNK | LIBR | NAME | PFIX | POSI | SCOP | SIZE | STAT | TYPE | WAIT

ACCS - Access To Common Memory Pool

Possible values:	READ/WRITE
Default value:	WRITE

This parameter determines how the common memory pool can be accessed. READ means the access is write-protected.

To be able to set ACCS=READ, the user ID must be authorized for the BS2000/OSD CSTMP macro in the user catalog (JOIN command with C-M=YES).

ADDR - Size of Common Memory Pool Address

Possible values:	*number* (>=0)
Default value:	none

This parameter determines the number of megabytes for the defined address of the common memory pool.

ALNK - Activate AUTOLNK Function

Possible values:	YES/NO
Default value:	YES

This parameter determines whether the AUTOLNK function of the dynamic binder loader (DBL) is activated.

LIBR - Load Library

Possible values:	*lll*/BLSLIB/CLASS-4
Default value:	none

This parameter determines from where the defined module is to be loaded.

lll	The name of the load library.
BLSLIB	The libraries with the link names BLSLIB and BLSLIB01 to BLSLIB99 are to be used.
CLASS-4	Module is loaded as subsystem in class 4 memory.

If the operand of parameter LIBR is not defined, only a common memory pool will be enabled (ENAMP+REQMP).

NAME - Common Memory Pool/Module Name

Possible values:	*name* (valid name of common memory pool or module)
Default value:	none

This parameter determines the name of the module and/or the name of the common memory pool. The maximum number of characters in a name is:

8 characters	module name (name of common memory pool); Natural buffer pool.
16 characters	all other common memory pools.

PFIX - Fixed Address

Possible values:	NO/YES
Default value:	NO

This parameter determines whether the common memory pool’s address should be fixed or not.

For a global Natural load pool, this parameter must be set to YES.

POSI - Position Relative To 16-MB Line

Possible values:	BELOW/ABOVE
Default value:	BELOW

This parameter determines the position of the common memory pool, which can be above or below the 16-MB line.

SCOP - Scope of Common Memory Pool

Possible values:	LOCAL/GROUP/GLOBAL
Default value:	GLOBAL

This parameter determines the scope of the common memory pool. For more information on the scopes of a common memory pool, see the description of the ENAMP macro in the BS2000/OSD documentation.

SIZE - Size of Common Memory Pool

Possible values:	*n*KB/*n*MB
Default value:	1MB

This parameter specifies the size of the common memory pool in KB or in MB.

STAT - Status of Common Memory Pool

Possible values:	GLOBAL/LOCAL
Default value:	LOCAL

This parameter determines the status of the common memory pool.

STAT will be ignored when the program CMPSTART runs.

TYPE - Type of Common Memory Pool

Possible values:	COM/EDT/MON/NAT/SRT/SWP
Default value:	none

This parameter determines the type of the common memory pool. Possible types are:

COM	Natural DCOM pool
EDT	Editor buffer pool
MON	Natural monitor pool (SYSMON)
NAT	Natural buffer pool
SRT	Sort buffer pool
SWP	Natural swap pool

WAIT - Enabling or Waiting of Common Memory Pool During Application Startup

Possible values:	NO/YES
Default value:	NO

This parameter determines during startup of an application

- whether the common memory pool is to be enabled at once or
- whether the common memory pool is to wait for a request from Natural and is enabled then.

WAIT will be ignored when the program CMPSTART runs.

ADDEND Macro

The macro ADDEND defines the end of macro ADDON’s definitions. There are no parameters for ADDEND.

Example of Assembling Macro BS2STUB

```
BS2STUBA BS2STUB PARMOD=31,PROGMOD=24 /* 31-BIT ADDRESSING MODE,
   LOAD 3GL
/* PROGRAMS BELOW
/*
+--------------------------------------------------------------------+
/* I DEFINE THE GLOBAL Natural LOAD POOL I
/*
+--------------------------------------------------------------------+
ADDON NAME=NATSHARE,STAT=GLOBAL /* COMMON MEMORY POOL NAME IS NATSHARE
/* COMMON MEMORY POOL STATUS IS GLOBAL
/*
+--------------------------------------------------------------------+
/* I DEFINE THE GLOBAL Natural BUFFER POOL I
/*
+--------------------------------------------------------------------+
ADDON NAME=BP230GA,TYPE=NAT,STAT=GLOBAL
/*
+--------------------------------------------------------------------+
/* I DEFINE THE LOCAL Natural SWAP POOL I
/*
+--------------------------------------------------------------------+
ADDON NAME=SWAP#NAT230#LA,TYPE=SWP,SIZE=16MB,STAT=LOCAL,POSI=ABOVE
ADDEND
END
```

SYSBPMT Utility - Overview

The utility SYSBPMT provides statistical information on the current status of the Natural buffer pool including buffer pool cache and on the objects currently in the buffer pool and buffer pool cache. With SYSBPMT, you can also

- adjust the Natural buffer pool to meet your requirements;
- make objects resident in the buffer pool by marking them in SYSBPMT.

The SYSBPMT utility is menu-driven, and you can use a function code, a direct command or a PF key to perform a specific function.

For a general description of the Natural buffer pool, refer to Natural Operations for Mainframes. The buffer pool is defined with the macro NTBPI in the Natural parameter module, or with the corresponding dynamic profile parameter BPI, both of which are also described in Natural Operations for Mainframes.

The following topics are covered:

- Invoking and Operating SYSBPMT describes how to invoke the utility and how to select its functions by using the main menu or the direct commands provided. In addition, information on maintenance of further buffer pools, the use of SYSBPM in a Sysplex environment and buffer pool type INSTANCE - DCOM support is included.
- Buffer Pool Statistics provides buffer-pool-related, object-independent statistics functions including hash table statistics.
- BP Cache Statistics describes the buffer pool (BP) cache statistics functions including general BP cache statistics, BP cache call statistics, BP cache hash statistics, and individual cache object statistics.
- Individual Object Statistics provides information on the objects currently contained in the buffer pool.
- Object Directory Information displays the full directory of an object currently contained in the buffer pool.
- Display Object Hexadecimally displays an object currently contained in the buffer pool in hexadecimal format.
- Delete Object from Buffer Pool describes how to delete one or more objects from the buffer pool.
- Blacklist Maintenance describes how to maintain a blacklist of Natural objects which are not to be loaded into the buffer pool or, if they are already in the buffer pool, are to be deleted.
- Preload List Maintenance describes how to maintain so-called "preload lists" where you can specify the names of Natural objects to be loaded into the buffer pool on its initialization.

Invoking and Operating SYSBPMT

This section describes how to invoke the SYSBPMT utility and how to select its functions by using the main menu or the direct commands provided.

In addition, information on maintenance of further buffer pools, the use of SYSBPMT in a Sysplex environment and buffer pool type INSTANCE - DCOM support is included.

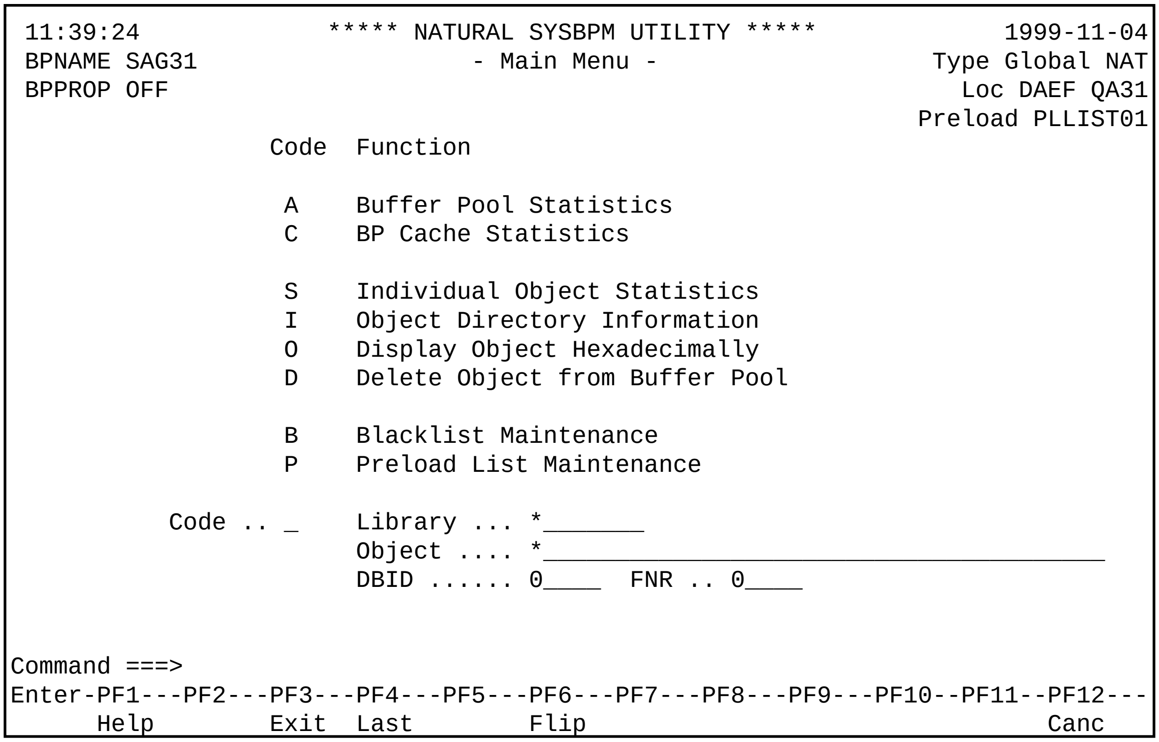
This section covers the following topics:

- Invoking SYSBPMT
- Direct Commands in SYSBPMT
- Maintenance of Further Buffer Pools
- SYSBPMT in a Sysplex Environment
- Buffer Pool Type INSTANCES - DCOM Support

Invoking SYSBPMT

To invoke the SYSBPMT utility

- Logon to library SYSBPMT and start the program MENU.
  The main menu of SYSBPMT is displayed:



```
 11:39:24             ***** NATURAL SYSBPM UTILITY *****              1999-11-04
 BPNAME SAG31                   - Main Menu -                    Type Global NAT
 BPPROP OFF                                                        Loc DAEF QA31
                                                                Preload PLLIST01
                  Code  Function

                   A    Buffer Pool Statistics
                   C    BP Cache Statistics

                   S    Individual Object Statistics
                   I    Object Directory Information
                   O    Display Object Hexadecimally
                   D    Delete Object from Buffer Pool

                   B    Blacklist Maintenance
                   P    Preload List Maintenance

           Code .. _    Library ... *_______
                        Object .... *_______________________________________
                        DBID ...... 0____  FNR .. 0____


Command ===>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help        Exit  Last        Flip                                Canc
```

To invoke a function in the SYSBPMT utility

- Enter the code character in the Code field or
  use a direct command. See Direct Commands in SYSMBPT.

You can select the following functions from the SYSBPMT main menu:

Function	**Explanation**
Buffer Pool Statistics	This function leads to the buffer pool statistics sub-menu. From the sub-menu, you can start buffer pool related, object independent statistics functions including hash table statistics.
Buffer Pool Cache Statistics	This function leads to the buffer pool cache statistics sub-menu. From the sub-menu, you can start functions for the buffer pool cache.
Individual Object Statistics	This function displays the entries currently loaded in the Natural buffer pool. Each entry can be accessed individually, and various subfunctions can be performed for each entry.
Object Directory Information	This function displays the full directory information of a specified object currently contained in the Natural buffer pool.
Display Object Hexadecimally	This function displays in hexadecimal format a specified Natural object that is currently in the buffer pool.
Delete Object from Buffer Pool	This function is used to delete one or more Natural objects from the buffer pool.
Blacklist Maintenance	This function is used to maintain a blacklist of Natural objects which are **not** to be executed.
Preload List Maintenance	This function is used to maintain preload lists. In a preload list, you can specify the names of Natural objects which are to be loaded into the buffer pool when the buffer pool is initialized.

Asterisk notation (*) is possible in the Library and Object fields wherever a selection of objects can be made.

If you specify a "0" as DBID or FNR, the specified object(s) will be selected regardless of their DBID and FNR. Any value other than "0" represents a valid DBID or FNR specification.

Direct Commands in SYSBPMT

Most SYSBPMT functions can also be invoked via direct commands. You enter a direct command in a command line on any SYSBPMT screen.

The following direct commands are available in SYSBPMT:

Command	Parameters	Function
ADD BLACKLIST	none	Invokes the function "Maintain Blacklist".
ADD SET	*library-name set-name*	Invokes the function "Add Object Set to Blacklist".
BLACKLIST	none	Invokes the "Blacklist Maintenance" menu.
BOTTOM	none	Scrolls to the end of a list.
CANCEL	none	Same as EXIT.
CHECK HASH CHECK HT	none	Invokes the function "Check hash table for consistency". This function returns the number of inconsistencies found. If there are any inconsistencies, rebuild the hash table using the REBUILD HASH command.
CLEAR	none	See Functions for the Objects Displayed.
CLOSE BPC	none	Invokes the function "Close buffer pool cache". The buffer pool runs without cache afterwards. You can restart the buffer pool cache using the INITIALIZE BPC command.

Command	Parameters	Function
CLOSE HASH CLOSE HT	none	Invokes the function "Close hash table support". The buffer pool runs without hashing afterwards. To enable the hash support again, use the REBUILD HASH command or initialize the buffer pool.
DELETE	*library-name object-name dbid fnr*	Invokes the function "Delete Object from Buffer Pool and Buffer Pool Cache".
DELETE BP	*library-name object-name dbid fnr*	Invokes the function "Delete Object from Buffer Pool only".
DELETE BPC	*library-name object-name dbid fnr*	Invokes the function "Delete Object from Buffer Pool Cache only".
DELETE ALL	none	Deletes all objects from the blacklist.
DELETE BLACKLIST	none	Invokes the function "Maintain Blacklist" from where you can delete blacklist entries.
DELETE SET	*library-name set-name*	Invokes the function "Delete Object Set from Blacklist".
DISPLAY ALL	none	Same as DISPLAY INDIVIDUAL * *.
DISPLAY BUFFERPOOL DISPLAY BP	none	See Display Buffer Pools.
DISPLAY BLACKLIST	none	Invokes the function "Maintain Blacklist".
DISPLAY CGENERAL	none	Invokes the function " General BP Cache Statistics"
DISPLAY CHASH	none	Invokes the function BP Cache Hash Table Statistics".
DISPLAY CINDIVIDUAL	*library-name object-name dbid fnr*	Invokes the function "Individual Cache Object Statistics".

Command	Parameters	Function
DISPLAY CLOAD	none	Invokes the function " BP Cache Call Statistics"
DISPLAY CSTATISTICS	none	Displays the "BP Cache Statistics Main Menu".
DISPLAY DIRECTORY	*library-name object-name dbid fnr*	Invokes the function "Object Directory Information".
DISPLAY FRAGMENTATION	none	Invokes the function "Buffer Pool Fragmentation".
DISPLAY FUNCTION	none	Invokes the function "Internal Function Usage".
DISPLAY GENERAL	none	Invokes the function "General Buffer Pool Statistics".
DISPLAY HASH DISPLAY HT	none	Invokes the function " Buffer Pool Hash Table Statistics".
DISPLAY INDIVIDUAL	*library-name object-name dbid fnr*	Invokes the function "Individual Object Statistics".
DISPLAY LOAD	none	Invokes the function "Buffer Pool Load/Locate Statistics".
DISPLAY OBJECT	*library-name object-name dbid fnr*	Invokes the function "Display Object Hexadecimally".
DISPLAY STATISTICS	none	Invokes the BP Statistics Main Menu.
EDIT PRELOADLIST	*list-name*	Invokes the function "Edit Preload List".
EDIT SET	*library-name set-name*	Invokes the function "Edit Object Set".
EXIT	none	Leaves the current function and displays the previous screen.
FDELETE	none	See Functions for the Objects Displayed.
FLIP	none	Switches the PF-key line.

Command	Parameters	Function
GENERATE PRELOADLIST	*list-name gen-library gen-objects*	Invokes the function "Generate Preload List from Buffer Pool".
GP	none	See Scrolling in the Displayed Object.
INITIALIZE	none, 1, 2, 4, 8, 12, 16	Re-initializes the buffer pool and the buffer pool cache. If no text record size is specified, the current text record size will be taken.
INITIALIZE BP	none, 1, 2, 4, 8, 12, 16	Re-initializes the buffer pool only. If no text record size is specified, the current text record size will be taken.
INITIALIZE BPC	none	Re-initializes the buffer pool cache only. To avoid program abends of other users, it is recommended to close the buffer pool cache before initializing it. The text record size of the buffer pool cache is fixed (4 KB).
INITIALIZE OLD	none, 1, 2, 4	Re-initializes the buffer pool with the old format. If no text record size is specified, the current text record size will be taken. This enables sessions with the parameter setting PLUGIN=NOBP to access this buffer pool (fallback).
KST	none	See Scrolling in the Displayed Object.
LAST	none	Displays the most recently entered SYSBPMT direct command.
LEFT	none	See Scrolling in the List.
LIST PRELOADLIST	*list-name*	Invokes the function "List Preload Lists".
LIST SET	*library-name set-name*	Invokes the function "List Object Sets".
MENU	none	Invokes the SYSBPMT main menu.

Command	Parameters	Function
MPT	none	See Scrolling in the Displayed Object.
NEXT	none	Displays the next item selected when several objects have been selected at a time (only available within the functions "Object Directory Information" and "Display Object Hexadecimally").
PRELOADLIST	none	Invokes the "Preload List Maintenance" menu.
QUIT	none	Same as EXIT.
REBUILD HASH REBUILD HT	none	Invokes the function "Rebuild Hash Table". Deletes the hash table and rebuilds it according to current buffer pool content.
REORG HASH REORG HT	none	This function is not yet implemented.
RESET BUFFERPOOL RESET BP	none	See Reset Buffer Pool.
RESIDENT	none	See Functions for the Objects Displayed.
RIGHT	none	See Scrolling in the List.

SELECT BUFFERPOOL
SELECT BP
	none	See Select Buffer Pool.

STOP	none	Leaves the SYSBPMT utility.
TOP	none	Scrolls to the beginning of a list.

UPDATE or
UPDATE BLACKLIST
	none	Corresponds to PF5 in the "Maintain Blacklist" function.

+	none	Scrolls one page down in a list.
-	none	Scrolls one page up in a list.

Maintenance of Further Buffer Pools

With the SYSBPMT utility, you can also maintain buffer pools other than the current one.

Three functions are provided for this purpose:

- Display Buffer Pools
- Select Buffer Pool
- Reset Buffer Pool

Display Buffer Pools

When you enter the direct command DISPLAY BUFFERPOOL, a window is displayed which contains the following information on your current (global or local) buffer pool and on all further *global* buffer pools currently available within your Natural system: the name of the buffer pool, the kind of buffer pool (Natural, Sort, DL/I, Editor, Monitor, Instance), the current status, the name of the preload list (if applicable) and the storage address.

Select Buffer Pool

When you enter the direct command SELECT BUFFERPOOL, a window is displayed with a list of all buffer pools.

The information displayed is the same as with the function "Display Buffer Pools (see above).

In the window, you can select a buffer pool by marking it with any character. Once you have selected a buffer pool from the window, all SYSBPMT functions apply to this buffer pool.

Your Natural session itself, however, will continue to run with the original buffer pool.

With the function "Reset Buffer Pool" (see below), you can switch SYSBPMT back to your original buffer pool.

Reset Buffer Pool

Once you have used the functions of SYSBPMT for another buffer pool (see "Select Buffer Pool" above), you switch the applicability of SYSBPMT back to the buffer pool used by your current Natural session.

To do so, you enter the direct command RESET BUFFERPOOL.

SYSBPMT in a Sysplex Environment

In the top right corner "Loc" entry of the SYSBPMT screen, the host ID and the subsystem ID are displayed.

Whenever Natural switches to another host or subsystem, the host ID or subsystem ID changes. Since the Natural buffer pool will not switch to the new host or subsystem, Natural will use a new buffer pool located in the new host or subsystem. Switching can take place after each terminal I/O, that is, after pressing any function key. After switching, browsing and positioning functions will not be executed (top, bottom, +, -, left, right). Instead, the displayed list starts from the top of the new buffer pool. If the BPPROP parameter is set to "PLEX" or to "GPLEX", operations on the buffer pool like "delete", "add" or "resident" are executed and are then propagated to the other available buffer pools.

If a function has been aborted or has been propagated, a corresponding message is issued. There is also a message each time Natural has switched to another host or subsystem and the buffer pool was changed ("Attention: BP switched").

Buffer Pool Type INSTANCES - DCOM Support

SYSBPMT also allows you to maintain an instances buffer pool containing objects used for DCOM operations. The instances buffer pool offers the same functions as the other Natural buffer pool types. However, for the DCOM class MODEL, the object name (A40) is a member name with format A8, and for the DCOM classes FACTORY/INSTANCE, the object name is a class name with format A32 and a server/instance ID with format A8.

SYSBPMT - Buffer Pool Statistics

This function leads to the buffer pool statistics submenu. From the submenu, you can start buffer pool related, object independent statistics functions including hash table statistics.

To select the statistics submenu

- enter code "A" in the code field of the SYSBPMT main menu,
  or enter the direct command DISPLAY STATISTICS.

The Buffer Pool Statistics Main Menu is then displayed, from which you can select the following functions:

- General Buffer Pool Statistics
- Buffer Pool Load/Locate Statistics
- Buffer Pool Fragmentation
- Internal Function Usage
- Buffer Pool Hash Table Statistics

General Buffer Pool Statistics

This function is used to monitor the performance of the buffer pool, and displays various statistics regarding the activity of the buffer pool.

You can either select this function on the Buffer Pool Statistics Main Menu, or invoke it by entering the direct command DISPLAY GENERAL.

The "General Buffer Pool Statistics" screen is then displayed. The statistics displayed on this screen are "snapshots" of the buffer pool which are refreshed each time you press ENTER. The following information is displayed:

Item	Explanation
Buffer Pool Address	Shows the storage address of the buffer pool; that is, of the buffer pool control block.
Directory Section	Shows the storage address of the buffer pool directory section (relative to the beginning of the buffer pool). Each object stored in the buffer pool requires a directory entry that contains information about this object. The space for these directory entries is acquired from the buffer pool itself.
Text Record Section	Shows the storage address of the text record section (relative to the beginning of the buffer pool). After the space used by the directory entries has been allocated, the remaining space is divided into blocks called text records (whose size, by default, is 4 KB). A Natural object can occupy one or more text records, depending on its size.
Dataspace attached	Shows the name of the attached dataspace for the BP cache.
Buffer Pool Size (MB)	Shows the size of the whole buffer pool in MB. The buffer pool size can be changed with the NTBPI macro in the parameter module or with the BPI profile parameter (as described in Natural Operations for Mainframes).
Directory Size	Shows the size of a directory entry in bytes.

Item	Explanation
Text Record Size	Shows the size of a text record in kbytes. The text record size can be changed with the NTBPI macro in the parameter module or with the BPI profile parameter (as described in Natural Operations for Mainframes). The default text record size is set to 4 KB; however, if you use applications that consist of many rather small objects, it is recommended that you reduce it to 2 KB. This reduces the percentage of unused space in the buffer pool, although it can lead to search algorithm 2 being invoked more frequently.
Initialisation	Shows the date and time the buffer pool was initialised.
Last Refresh	Shows the date and time the buffer pool was most recently refreshed, and the ID of the user who performed the refresh.
Text Records - Total	Shows the total number of text records.
Text Records - Used	Shows the number of text records currently used.
Text Records - Used (%)	Shows the percentage of currently used text records.
Text Records - Max Used	Shows the maximum number of used text records.
Text Record - Total Size	Shows the total text record size, which is the number of active text records times the single text record size. The difference between the Total Text Record Size and the Total Object Size shows the amount of unused size in the text record section and can also be an indicator for the system administrator of whether to modify the text record size or not.
Text Record - Avg Usage (%)	Shows the average usage of text records in percent. This value should not be significantly less than 75%. If the buffer pool is almost full, any value above 75% indicates a good usage of the buffer pool. If the usage is significantly less than 75%, the text record size should be reduced.

Item	Explanation
Space Used (%)	Shows the actual usage of the buffer pool space in percent. Note: If the buffer pool is almost full (that is, the value in the Text Records Used field - see above - is almost 100%), any value above 75% indicates a good usage of the buffer pool. If the usage is significantly less than 75%, the text record size should be reduced.
Objects - Loaded	Shows the number of objects currently loaded in the buffer pool.
Objects - Max Loaded	Shows the maximum number of objects loaded in the buffer pool.
Objects - Total Size	Shows the total size of the objects currently loaded in bytes.
Objects - Avg TR Used	Shows average number of text records used by one object.
SumOfUseCounts	Sums up the use counts of all objects currently loaded in the buffer pool. A use count counts all current users of a given object. If an object is currently not in use, its use count returns to 0.
AvgLifetimeUsed (min)	Shows the average load time (in min) of objects currently loaded in the buffer pool.
AvgLifetimeReplace (min)	Shows the average load time (in min) of objects, which have already been replaced, i.e. deleted in the buffer pool.

Buffer Pool Load/Locate Statistics

This function provides statistical information on the loading of objects into the buffer pool and the locating of objects in the buffer pool. This information also serves as buffer pool performance indicator.

You can either select this function on the Buffer Pool Statistics Main Menu, or invoke it by entering the direct command DISPLAY LOAD.

The "Buffer Pool Load/Locate Statistics" screen is then displayed. The statistics displayed on this screen are "snapshots" of the buffer pool which are refreshed every time you press ENTER. The following information is displayed:

Information	Explanation
Total Locate Calls	Shows the total number of program location calls; that is, the total number of times the Natural buffer pool manager was requested to search the buffer pool for an object. If the location is successful, the object has been loaded from the buffer pool or the cache and need not be loaded from a Natural system file thereby saving calls and I/Os.
Total Locate Calls successful	Shows the total number of successfully performed locate calls in absolute figures.
Total Locate Calls failed	Shows the total number of locate calls that failed.
Quick Locates	Shows the total number of quick locate requests. Quick location means that the directory address of the last call of the requested program is still available. This is due to the fact that Natural maintains user-specific tables of internal directory entries which contain information about the most recently used objects by each Natural user. When a user invokes an object that has been used before in the Natural session, Natural passes this information to the buffer pool manager, which then bypasses the normal locate procedure. If the last call address cannot be found, a normal locate request is automatically scheduled by the buffer pool manager.

Information	Explanation
Quick Locates successful	Shows the number of quick locate requests which have been successfully performed.
Quick Locates failed	Shows the number of quick locate requests which failed. Failed quick locates result in Normal Locate calls.
Normal after Quick	Shows the number of normal locate calls that have been preceded by a quick locate call.
Normal after Quick successful	Shows the number of normal locate requests that have been successful in locating the required Natural object in the buffer pool or the cache and have been preceded by a quick locate call.
Normal after Quick failed	Shows the number of normal locate calls that failed and have been preceded by a quick locate call.
Normal Locates	Shows the total number of normal locate requests.
Normal Locates successful	Shows the number of normal locate requests that have been sucessful in locating the required Natural object in the buffer pool.
Normal Locates failed	Shows the number of normal locate requests which failed. A failed normal locate request indicates that a program has to be loaded from the database or from the buffer cache.
Locates from cache	Shows the total number of successful locates of objects that resided in the buffer pool cache. This information is counted only if the previous Locate ("Normal after Quick failed" or "Normal Locates failed") failed. It indicates the number of database loads saved. This means, that, without the buffer pool cache, the object would have to be loaded from the database.
Load Calls (DB)	Shows the total number of load calls made since the buffer pool has been refreshed.
Program Loads	Shows the number of times a Natural object was loaded from a Natural system file into the buffer pool. As several load calls may be necessary to load a single object, this value provides the actual number of program loads made since the most recent buffer pool refresh.

Information	Explanation
Finished Loads	Shows how many of the program loads above have been successfully executed. For example, a load call could have failed, because at the time of the load, all objects in the buffer pool were currently being executed. When loading a Natural object, the buffer pool manager uses two search algorithms (see below).
Concurrent Loads	Shows the number of loads that have been done to load an object, which is being loaded already by another application. This means, that the load of an object is not yet finished, but another application requested the same object and the object will be loaded more than once.
Algorithm 1	Shows the number of times a storage allocation request was satisfied by Search Algorithm 1. The storage allocation request may be triggered either by a load from the database or by a load from the buffer pool cache. Search Algorithm 1 attempts to find a single piece of storage in the buffer pool or a single object that is not currently being executed and can be deleted, in order to obtain the required space to accomplish the load. If an object has to be deleted, Algorithm 1 compares the time stamps of the objects in question and the oldest one will be deleted. The search for space begins at the top of the text record section. If storage is unavailable after Algorithm 1 completes its search, Algorithm 2 is invoked.

Information	**Explanation**
Algorithm 2	Shows the number of times a storage allocation request was satisfied by Search Algorithm 2. The storage allocation request may be triggered either by a load from the database or by a load from the buffer pool cache. Search Algorithm 2 is a more expensive search than is Algorithm 1. Algorithm 2 attempts to combine two or more entities (free storage and/or objects not being executed) in order to obtain the necessary storage for the load. The ages of the objects, however, are not taken into account. Algorithm 2 begins its search at a point in the text record section where it last left off doing such a search, continues to the bottom of the text record section and, if necessary, wraps around to the top of the text record section to make one final pass from the top to the bottom. If space is still unavailable, Algorithm 2 fails, the object cannot be loaded and a corresponding error message is returned.
Largest Allocation	Shows the largest single allocation size so far requested, specified in number of text records.
Allocation Failure	Shows the total number of times an object load failed. The reason for a failure is that either all directory entries are in use at the time of the load request or not enough storage is available in the text record section to perform the load.
Last Failing Sizes	Shows the number of text records that would have been required by the three most recent storage allocation requests that failed. Storage allocation request failures are a direct result of search algorithm 2 failing for an object load request.
Total Locate Calls successful/Program Loads	This statistic is expressed as a ratio using "Total Locate Calls successful" and "Program Loads". A value greater than 1 indicates that Natural located more objects in the buffer pool than it loaded from the system file. This ratio serves as a buffer pool efficiency indicator. The larger the number, the better the buffer pool is performing. This is the *primary indicator* of performance from one buffer pool session to the next.

Buffer Pool Fragmentation

This function provides an overview of the buffer pool fragmentation; that is, an overview of how many different objects occupy how many text records, and how the object locations are spread over the buffer pool.

You can either select this function on the Buffer Pool Statistics Main Menu, or invoke it by entering the direct command DISPLAY FRAGMENTATION.

The "Buffer Pool Fragmentation" screen is then displayed. It provides the following information:

- Buffer Pool Size
- Buffer Pool Address
- Text Record Section
- Text Record Size
- Number of Text Records

In addition, the screen displays a diagram which shows how many different individual objects occupy how much text record size. For example:

```
          1---+----10---+----20---+----30---+----40---+----50
005F0480  .._.+***______++  ....**_.+**_.++_ *..+**+__++++XX
```

Each symbol in the diagram represents one text record, and each sequence of equal symbols represents a different individual object occupying one or more text records. The symbols have the following meaning:

_ and .	Objects with a use count of 0.
+ and *	Objects with a use count greater than 0.
blank	An unused text record.
X	The end of the buffer pool, which means that no further text records are available.

In the example above, the buffer pool contains 48 text records. Three of them are not in use; the rest is occupied by 24 different Natural objects, 12 of them with a use count of 0, and 12 with a use count greater than 0.

Internal Function Usage

This function provides statistical information on the calls to the Natural buffer pool manager.

You can either select this function on the Buffer Pool Statistics Main Menu, or invoke it by entering the direct command DISPLAY FUNCTION.

The "Internal Function Usage" screen then is displayed.

The statistics displayed on that screen are "snapshots" of the buffer pool which are refreshed every time you press ENTER.

The field "Total Calls" shows the overall number of all internal calls that have been made to the buffer pool manager.

Internally, the buffer pool manager can be invoked for various different functions. For each function, the number of times it has been invoked is displayed, both as an absolute figure and as a percentage figure. In addition, these figures are represented in a horizontal bar chart.

Buffer Pool Hash Table Statistics

This function displays statistics about hash table slots and collisions per slot. The statistics determine the efficiency of the hash algorithm used.

You can either select this function on the Buffer Pool Statistics submenu, or invoke it by entering the direct command DISPLAY HASH.

The "Hash Table Collisions" screen is then displayed. The statistics displayed on this screen are "snapshots" of the hash table which are taken every time you press ENTER. The following information is displayed:

Information	Explanation
Total number of slots	Shows the total number of hash table slots; that is, the total possible entries that link the object name with the location of the object. The number of slots i.e. the size of the hash table will be calculated internally depending on the number of text records.
Number of slots used	Shows the number of slots that have one or more entries.
Number of slots free	Shows the number of slots that have no entry.
Max. Collisions per slot	Shows maximum number of collisions of all slots. The maximum number of collisions is the longest possible search path for an object.
Collisions	The number of possible collisions. 0 means no collision or one entry. When there are more than 5 collisions, the number of collisions will be specified in ranges (e.g. 6 - 10).
Number of Slots	Shows the number of slots grouped by their number of collisions. For example, if the number of collisions is 3, the search algorithm must side step a maximum of 3 times to find an object. In addition the percentage of these slots related to all slots used is displayed.
Totalled Number of Slots	Shows the same values as Number of Slots, but the values are summed up.

SYSBPMT - BP Cache Statistics

This function leads to the buffer pool cache statistics sub-menu. From the sub-menu, you can start the function for the buffer pool cache.

To start the function

- enter code "C" in the code field of the SYSBPMT main menu,
  or enter the direct command DISPLAY CSTATISTICS.

The BP Cache Statistics Main Menu is then displayed, from which you can select the following functions:

- General BP Cache Statistics
- BP Cache Call Statistics
- BP Cache Hash Table Statistics
- Individual Cache Object Statistics

General BP Cache Statistics

This function displays various addresses and statistics regarding the activity of the buffer pool cache.

You can either select this function on the BP Cache Statistics main menu, or invoke it by entering the direct command DISPLAY CGENERAL.

The "General BP Cache Statistics" screen is then displayed. The statistics displayed on this screen are "snapshots" of the buffer pool, which are refreshed each time you press ENTER. The following information is displayed:

Item	Explanation
Dataspace Name	Shows the name of the dataspace where the BP cache resides.
Dataspace SToken	The term SToken (for "Space Token") identifies a dataspace.
Dataspace ALET	The term ALET (for "Address List Entry Token") identifies an index for accessing the dataspace.

Item	Explanation
Dataspace size (MB)	Shows the size of the buffer pool cache in MB.
Dataspace Current state	Shows the state of the cache. The state may be "not initialized", "locked for init", "closed", "free for operation" or "undefined".
Dataspace Initialisation	Shows the date and time when the buffer pool cache was initialised.
Internal buffer offsets	● Header Buffer: header of the cache, contains general cache information ● Hash Buffer: contains the hash table ● Directory Buffer: Shows the storage address of the BP cache directory section (relative to the beginning of the buffer pool cache). Each object stored in the buffer pool cache requires a directory entry that contains information about this object. The space for these directory entries is acquired from the buffer pool cache itself. ● Text Buffer: Shows the storage address of the text buffer (relative to the beginning of the buffer pool cache). After allocating the space for all other buffers, the remaining space is divided into text records with a size of 4 KB. A Natural object can occupy one or more text records, depending on its size.
Tot. Text Records	Shows the total number of text records in the BP cache. The number of text records depends on the cache size. The text record size for the cache is 4KB.
Insert Position	Shows the index number of the text record into which the next object will be inserted. Objects will be inserted into the cache when they have to be removed from the buffer pool.
Re-use cycles	Shows the number of times the cache has been completely reused. Every time the cache is full, the cache manager re-uses the cache from the start and overwrites the object(s) from there. The objects will remain in the cache until the cache is re-used again.
Objects - Max Loaded	Shows the maximum number of objects loaded in the buffer pool cache.
Objects - Loaded	Shows the number of objects currently loaded in the buffer pool cache.

BP Cache Call Statistics

This function provides statistical information on the loading (put), retrieving (get) and deleting of objects into/from the buffer pool cache. This information also serves as a buffer pool cache performance indicator.

You can either select this function on the BP Cache Statistics main menu, or invoke it by entering the direct command DISPLAY CLOAD.

The "BP Cache Call Statistics" screen is then displayed. The statistics displayed on this screen are "snapshots" of the buffer pool which are refreshed each time you press ENTER. The following information is displayed:

Item	Explanation
Search calls (total)	Shows the total number of search calls issued from the buffer pool to the buffer pool cache. If an object was found, a search call results into a get call.
Get calls (from cache)	The buffer pool issues a get call to load an object from the cache into the buffer pool.
Get calls successful	An object that was requested by the buffer pool was really loaded from the cache into the buffer pool. A get call may be unsuccessful, e.g. if an object has been deleted after is was found by the search call.
Put calls successful	An object was really put from the buffer pool to the cache.
Obj. already cached	Shows the number of object which the buffer pool wanted to put to the cache, but the objects have already been in the cache, i.e. cached before.
Delete Calls	Shows the number of delete calls to the cache.
Delete Calls successful	Shows the number of successful delete calls to the cache. A delete call is unsuccessful if the object to be deleted is not in the cache.
Get/Put Rate	Shows how many "gets" have been successfully related to one successful "put". The value will be calculated by dividing the successful "gets" by the successful "puts". The higher this value, the better is the cache efficiency.

Item	Explanation
Get/Search Rate	Shows the number of searches for objects (in percent), which could successfully be served by a "get" of an object from the cache. The higher this value, the better is the cache efficiency.
Initialisation	Shows the date and time the buffer pool cache was initialised.
Last re-use cycle	Shows the load date and time of the last object that has been overwritten. An object will be overwritten when its space has to be re-used. This means the load time of the object which has been in the cache longest corresponds to the "Last re-use cycle" date and time.
Last access	Shows the date and time of the last access to the cache.
Last PUT (to cache)	Shows the date and time of the last put call.
Last GET (from cache)	Shows the date and time of the last get call.
Last DELETE	Shows the date and time of the last delete call.

BP Cache Hash Table Statistics

This function displays statistics about hash table slots and collisions per slot. The statistics determine the efficiency of the hash algorithm used.

You can either select this function on the BP Cache Statistics main menu, or invoke it by entering the direct command DISPLAY CHASH.

The "Cache Hash Table Collisions" screen is then displayed. The statistics displayed on this screen are "snapshots" of the hash table which are refreshed every time you press ENTER. The following information is displayed:

Information	Explanation
Total number of slots	Shows the total number of hash table slots; that is, the total possible entries that link the object name with the location of the object. The number of slots i.e. the size of the hash table will be calculated internally depending on the number of text records.
Number of slots used	Shows the number of slots that have one or more entries.
Number of slots free	Shows the number of slots that have no entry.
Max. collisions per slot	Shows the maximum number of collisions of all slots. The maximum number of collisions is the longest possible search path for an object.
Collisions	The number of possible collisions. 0 means no collision or one entry. When there are more than 5 collisions, the number of collisions will be specified in ranges (e.g. 6 - 10).
Number of Slots	Shows the number of slots grouped by their number of collisions. For example, if the number of collisions is 3, the search algorithm must side step a maximum of 3 times to find an object. In addition, the percentage of these slots related to all slots used is displayed.
Totalled Number of Slots	Shows the same values as Number of Slots, but the values are summed up.

Individual Cache Object Statistics

This function shows you the objects currently loaded in the buffer pool cache. Each object can be deleted individually.

You can either select this function on the BP Cache Statistics main menu, or invoke it by entering the direct command DISPLAY CINDIVIDUAL.

The "Individual Cache Object Statistics" screen is then displayed, showing a list of all individual objects currently contained in the buffer pool cache.

The statistics displayed on this screen are "snapshots" of the contents of the buffer pool which are refreshed every time ENTER is pressed.

For each object, the following information is displayed:

Item	Explanation
C	In this column, you can enter a command to perform a function for the object. For the buffer pool cache, only the command "DE" (delete object from BP cache) is allowed.
Library	The library from which the object was loaded.
Object	The name of the object.
DBID	The database ID of the Natural system file from which the object was loaded.
FNR	The file number of the Natural system file from which the object was loaded.
L	● L: Locked for load ● G: Locked for get ● D: Locked for delete ● X: Locked
Reuse	Reuse counter of an object: Indicates how often an object has been loaded (reused) from the cache into the buffer pool.
BPTotalUC	Buffer Pool Total Use Counter: Indicates how often an object has been used by a Natural application from the buffer pool since the object had been loaded from the database.
Size	The size of the object.
BPCSize	The size that has to be allocated in the buffer pool cache to contain the object. This is a multiple of the text record size (4 KB).
Message (Msg)	The message in response to your command entry in column "C". Possible messages are: ● "failed" (in response to any function whose execution was not successful), ● "deleted" (in response to commands DE).

Scrolling in the List

To scroll in the list, you can use the following PF keys and direct commands:

Key	Command	Function
PF7	-	Scrolls one page up.
PF8	+	Scrolls one page down.
PF19	TOP	Scrolls to the beginning of the list.
PF20	BOTTOM	Scrolls to the end of the list.

SYSBPMT - Individual Object Statistics

This function shows you the objects currently in the buffer pool. Each object can be accessed individually, and various functions can be performed for each object.

You can either select this function on the SYSBPMT main menu, or invoke it by entering the direct command DISPLAY INDIVIDUAL.

The "Individual Object Statistics" screen is then displayed, showing a list of all individual objects currently contained in the buffer pool.

The statistics displayed on this screen are "snapshots" of the contents of the buffer pool which are refreshed every time ENTER is pressed.

For each object, the following information is displayed:

Item	Explanation
C	In this column, you can enter a command to perform a function for the object. The functions are described below.
Library	The library from which the object was loaded.
Object	The name of the object. If the name is "RUNTIME", the object has been invoked with the RUN command.
DBID	The database ID of the Natural system file from which the object was loaded.
FNR	The file number of the Natural system file from which the object was loaded.
RLD	R The object is set to be resident in the buffer pool. L The object is currently being loaded into the buffer pool. D The object is pending a delete request. It is deleted from the buffer pool as soon as the value in the Use column returns to 0.
Use	The number of Natural users who are executing this object at this specific point in time.

Item	Explanation
Max	The maximum number of users who have been executing this object concurrently since it has been loaded into the buffer pool.
Total	The total number of locate calls of an object since it has been loaded into the buffer pool. If a cache is used, this value is not lost if the object is removed from the buffer pool and saved to the cache. Therefore, this value indicates the number of uses of this object since it has been loaded from the system file.
Size	The size of the object.
BP Size	The size that has to be allocated in the buffer pool to contain the object.
Message (Msg)	The message in response to your command entry in column "C". Possible messages are: "failed" (in response to any function whose execution was not successful), "deleted" (in response to the command DE or FO), "released" (in response to the command CL), "resident" (in response to the command RE).

Scrolling in the List

To scroll in the list, you can use the following PF keys and direct commands:

Key	Command	Function
PF7	-	Scrolls one page up.
PF8	+	Scrolls one page down.
PF10	LEFT	Shows default display (*).
PF11	RIGHT	Shows full object names and messages (*).
PF19	TOP	Scrolls to the beginning of the list.
PF20	BOTTOM	Scrolls to the end of the list.

* By default, "Object" and "Message" are displayed in abbreviated form. With PF11, you display both fields in full length. With PF10, you switch back to default display.

Functions for Individual Objects

For each object on the list, you can perform one of the following functions by entering one of the following commands (in column "C"):

Command	Function
CL	Releases an object marked as resident.
LD	Corresponds to the function "Object Directory Information".
FO	Deletes an object immediately from the buffer pool, regardless of its use count.
DE	Marks an object to be deleted from the buffer pool. The object is deleted as soon as its use count becomes "0".
RE	Marks an object as resident, which means that the object is not deleted from the buffer pool, not even if its use count becomes "0".
OB	Corresponds to the function "Display Object Hexadecimally".

If you press PF1, a list of the available commands and functions is displayed.

SYSBPMT - Object Directory Information

This function displays the full directory of an object currently contained in the buffer pool.

You can either select this function on the SYSBPMT main menu, or invoke it by entering the direct command DISPLAY DIRECTORY.

The "Object Directory Information" screen is then displayed. The screen is divided into three sections:

The upper section of the screen shows the name, library name, database identification and file number of the specified object. In addition, it shows when, and requested by which user, the object was loaded into the buffer pool, and the date and time of the most recent action performed on the object.

The middle section of the screen provides the following information:

Item	Explanation
BP Directory at Address	The storage address of the directory of the specified object in the buffer pool.
Object at Address	The storage address of the specified object in the buffer pool.
Allocated Size	The size that has to be allocated to contain the specified object. It is a multiple of the text record size.
Object Size	The size of the specified object.
Status	R The object is set to be resident in the buffer pool. L The object is currently being loaded into the buffer pool. D The object is pending a delete request. It is deleted from the buffer pool as soon as its current use count becomes 0.
Current Use Count	The number of Natural users who are executing this object at this specific point in time.
Maximum Use Count	The maximum number of concurrent executes of this object since it has been loaded into the buffer pool.
Total Use	The total number of executes of this object since it has been loaded into the buffer pool. If a cache is used, this value is not lost if the object is removed from the buffer pool and saved to the cache. Therefore, this value indicates the number of uses of this object since it has been loaded from the system file.

The information shown in the lower section of the screen is similar to the information you receive with the Natural system command LIST DIRECTORY.

Functions for the Objects Displayed

For the object displayed, you can perform one of the following functions by entering one of the following direct commands in the command line:

Command	Function
FDELETE	Deletes an object immediately from the buffer pool, regardless of its use count.
RESIDENT	Marks an object as resident, which means that the object is not deleted from the buffer pool, not even if its use count becomes "0".
CLEAR	Releases an object marked as resident.

SYSBPMT - Display Object Hexadecimally

This function displays an object currently contained in the buffer pool in hexadecimal format.

You can either select this function on the SYSBPMT main menu, or invoke it by entering the direct command DISPLAY OBJECT.

The "Display Object Hexadecimally" screen is then displayed, showing the object in hexadecimal format.

Scrolling in the Displayed Object

Within the displayed object, you can move to a specific location by entering either an absolute hexadecimal address or a hexadecimal offset relative to your current position.

With PF7, you can scroll one page backward; with PF8, one page forward.

In addition, you can enter the following positioning commands in the command line of this screen:

Command	Moves To
GP	The generated program.
KST	The constant table.
MPT	The multiple-purpose table.

SYSBPMT - Delete Object from Buffer Pool

With this function, you can delete one or more objects from the buffer pool. Unless specified differently (see Direct Commands), an object will alway be deleted from the buffer pool cache too.

To delete objects from the buffer pool

- Select this function on the SYSBPMT main menu or
  enter the direct command DELETE.
- Specify the object(s) to be deleted either in the corresponding input fields on the SYSBPMT main menu or
  as parameters following the direct command.
- In the Library and Object fields, you can specify a fully-qualified name, a partly-qualified name or asterisk notation (*).
- In addition, you can specify a DBID and FNR as selection criteria. Default values are "0"; that is, no specification of a database and/or file number is made.

Deleting a Single Object

To delete a specific object from the buffer pool

- Specify the full name of the object and the library, as well as the DBID and FNR.

Deleting All Objects of a Library

To delete from the buffer pool all objects that were loaded from a specific library

- Specify the library name, DBID and FNR; in the Object field, you enter an asterisk (*).

Deleting All Objects

To delete all objects currently in the buffer pool

Enter an asterisk (*) in the Library field and in the Object field, and enter a "0" in the DBID and FNR fields.

Time of Deletion

Objects that have a current use count of 0 are deleted immediately. Objects that have a non-zero current use count are deleted as soon as their current use count returns to 0.

SYSBPMT - Blacklist Maintenance

This function is used to maintain a blacklist of Natural objects which are not to be loaded into the buffer pool or, if they already are in the buffer pool, are to be deleted. If the cache is enabled, the objects will also be deleted from the cache.

The blacklist always applies to the currently active buffer pool.

You can add/delete individual objects to/from the blacklist (with the first function listed below). You can also create *object sets*: In such a set, you specify the objects not to be executed, and then only have to add a single set (instead of multiple individual objects) to the blacklist. You can also combine both, that is, you can add/delete objects to/from the blacklist individually as well as by sets.

An object set is stored as a Natural object of type "text".

For details on the blacklist, see the description of the Natural buffer pool in Natural Operations for Mainframes.

You can either select the Blacklist Maintenance function on the SYSBPMT main menu, or invoke it by entering the direct command BLACKLIST.

The Blacklist Maintenance menu is then displayed, from which you can select the following functions:

- Maintain Blacklist
- List Object Sets
- Edit Object Set
- Add Object Set to Blacklist
- Delete Object Set from Blacklist

See also Blacklist Maintenance in Batch Mode.

Maintain Blacklist

This function is used to display the current blacklist, and add/delete objects to/from it.

You can either select this function on the Blacklist Maintenance menu, or invoke it by entering one of the direct commands ADD BLACKLIST, DISPLAY BLACKLIST or DELETE BLACKLIST.

The current blacklist, that is, a list of all objects currently on the blacklist, will be displayed.

Within the blacklist, you can scroll backwards or forwards by pressing PF7 or PF8 respectively.

By default, the "Maintain Blacklist" function is invoked in "Display Mode" (for displaying the blacklist and modifying and deleting entries). For adding objects to the blacklist, you use "Add Mode". To switch from one mode to the other, you use PF9.

Adding Objects to the Blacklist

To add objects to the current blacklist, you specify their library and object names along with the corresponding DBIDs and FNRs on this screen. Then you press PF5 to confirm the addition.

Deleting Individual Objects from the Blacklist

To delete an object from the blacklist, you mark it with the command "DE" in the first column of the screen. When you press ENTER, the marked objects will be deleted (and corresponding messages will be displayed).

Deleting All Objects from the Blacklist

To delete the entire blacklist, you press PF2. A window is then displayed in which you confirm the deletion by entering "Y". (Should you decide against the deletion, either press PF3 without entering anything in the window or enter "N".)

List Object Sets

This function displays a list of all defined object sets.

You can either select this function on the Blacklist Maintenance menu, or invoke it by entering the direct command LIST SET.

In the first column of the list, you can mark an object set with a command to edit or delete the object set or add/delete it to/from the blacklist. For a list of possible commands, you enter a question mark (?) in the first column.

Further commands can be entered in the command line. For a list of the commands available, you enter a question mark (?) in the command line.

Edit Object Set

This function is used to create a new object set and to add/delete objects to/from an existing object set.

You can select this function on the Blacklist Maintenance menu, invoke it from the List Object Sets screen (see above), or invoke it by entering the direct command EDIT SET.

To create a new object set, you specify the function code and a library, but no object set name, on the Blacklist Maintenance menu. Within a new object set, you must enter library and object name. If DBID and FNR are left blank, they will be taken from the current FUSER and FNAT in the SYS-prefixed libraries. To modify an existing object set, you specify its name and library along with the function code on the Blacklist Maintenance menu.

A screen will be displayed on which you can add/delete objects to/from the object set. The editing functions provided on this screen are a subset of the functions provided by the Software AG Editor.

For a list of the editor commands available, you enter a question mark (?) in the command line. For a list of the line commands available, you enter a question mark (?) in the first column of the screen.

Add Object Set to Blacklist

This function is used to add all objects of an object set to the blacklist.

You can select this function on the Blacklist Maintenance menu, invoke it from the List Object Sets screen (see above), or invoke it by entering the direct command ADD SET.

When you invoke this function, all objects of the specified object set will be added to the blacklist.

Delete Object Set from Blacklist

This function is used to delete all objects of an object set from the blacklist.

You can select this function on the Blacklist Maintenance menu, invoke it from the List Object Sets screen (see above), or invoke it by entering the direct command DELETE SET.

When you invoke this function, all objects of the specified object set will be deleted from the blacklist.

BPMBLBAT - Blacklist Maintenance in Batch Mode

Online, you can lock individual objects against being executed by using the Blacklist Maintenance functions as described above.

In batch mode, you do this by using a Natural batch job that uses the program BPMBLBAT in library SYSBPMT.

The program BPMBLBAT is used as follows:

- Start a Natural batch job in the usual way.
- The CMSYNIN file instructs the Natural nucleus to log on to SYSBPMT and to execute the program BPMBLBAT.
- The next command from CMSYNIN is the FIN command.
- BPMBLBAT reads the input from the CMOBJIN file, where the first card must be in either of the following two formats:

Format 1: **FUNC=LOCK, BPNAME=***name***, LIB=***name***, DBID=***nnn***, FNR=***nnn*

Format 2: **FUNC=RLS, BPNAME=***name*

Format 1 causes objects to be put on the blacklist of the specified buffer pool (BPNAME=*name*) The following cards must contain in Positions 1 to 8 the name of the object to be locked. For each card, an entry is added to the blacklist with the specified object name and the corresponding library name, DBID and FNR. The last card must contain a period (.) to indicate the end of the input.

Format 2 causes the blacklist to be deleted, which means that no objects are locked in the specified buffer pool.

Example 1:

This JCL is an example of how to lock programs A, B and C in buffer pool V23GBP:

```
//SAGBAT     JOB  ,T.TEST,CLASS=K,MSGCLASS=X,REGION=2048K
//*
//NATURAL    EXEC PGM=NAT220OBT,PARM='IM=D,OBJIN=Y'
//STEPLIB    DD   DSN=OPS.SYSF.TESTNAT.LOAD,DISP=SHR
//           DD   DSN=OPS.SYSF.V5.ADALOD,DISP=SHR
//DDCARD     DD   *
ADARUN PROGRAM=USER,SVC=249,DATABASE=10,MODE=MULTI
//SYSOUT     DD   SYSOUT=X
//SYSUDUMP   DD   SYSOUT=X
//CMSYNIN    DD   *
LOGON SYSBPMT
BPMBLBAT
FIN
//CMOBJIN    DD   *
FUNC=LOCK,BPNAME=V23GBP,LIB=SAGTEST,DBID=10,FNR=32
A
B
C
.
//CMPRINT    DD   SYSOUT=X
//
```

Example 2:

This JCL is an example of how to set to "0" the number of locked entries in buffer pool V23GBP:

```
//SAGBAT     JOB  ,T.TEST,CLASS=K,MSGCLASS=X,REGION=2048K
//*
//NATURAL    EXEC PGM=NAT220OBT,PARM=’IM=D,OBJIN=Y’
//STEPLIB    DD   DSN=OPS.SYSF.TESTNAT.LOAD,DISP=SHR
//           DD   DSN=OPS.SYSF.V5.ADALOD,DISP=SHR
//DDCARD     DD   *
ADARUN PROGRAM=USER,SVC=249,DATABASE=10,MODE=MULTI
//SYSOUT     DD   SYSOUT=X
//SYSUDUMP   DD   SYSOUT=X
//CMSYNIN    DD   *
LOGON SYSBPMT
BPMBLBAT
FIN
//CMOBJIN    DD   *
FUNC=RLS,BPNAME=V23GBP
//CMPRINT    DD   SYSOUT=X
//
```

SYSBPMT - Preload List Maintenance

This function is used to maintain so-called "preload lists". In a preload list, you can specify the names of Natural objects which are to be loaded into the buffer pool when the buffer pool is initialized.

The preload lists themselves are stored as objects of type "text" in the library SYSBPMT.

For details on preload lists, see the description of the Natural buffer pool in Natural Operations for Mainframes.

You can either select the Preload List Maintenance function on the SYSBPMT main menu, or invoke it by entering the direct command PRELOADLIST.

The Preload List Maintenance menu is then displayed, from which you can select the following functions:

- List Preload Lists
- Edit Preload List
- Generate Preload List from Buffer Pool

List Preload Lists

This function displays a list of all existing preload lists.

You can either select this function on the Preload List Maintenance menu, or invoke it by entering the direct command LIST PRELOADLIST.

In the first column of the list, you can mark a preload list with a command to edit or delete it. For a list of possible commands, you enter a question mark (?) in the first column.

Further commands can be entered in the command line. For a list of the commands available, you enter a question mark (?) in the command line.

Edit Preload List

This function is used to create a new preload list and to add/delete objects to/from an existing preload list.

You can select this function on the Preload List Maintenance menu, invoke it from the List Preload Lists screen (see above), or invoke it by entering the direct command EDIT PRELOADLIST.

To create a new preload list, you specify the function code, but no preload list name, on the Preload List Maintenance menu. Within a new preload list, enter library and object name. If DBID and FNR are left blank, they will be taken from the current FUSER and FNAT in the SYS-prefixed libraries. The resident flag will be set to "Y" in column "R" on the editing screen if no value is entered. To modify an existing preload list, you specify its name along with the function code on the Preload List Maintenance menu.

A screen will be displayed on which you can add/delete objects to/from the preload list. The editing functions provided on this screen are a subset of the functions provided by the Software AG Editor.

For a list of the editor commands available, you enter a question mark (?) in the command line. For a list of the line commands available, you enter a question mark (?) in the first column of the screen.

Generate Preload List from Buffer Pool

With this function, you can generate a new preload list using the object contained in the buffer pool; that is, from the object that are currently in the buffer pool, you can select those you wish to be included in the preload list.

You can select this function on the Preload List Maintenance menu or invoke it by entering the direct command GENERATE PRELOADLIST.

With the specifications in the fields "Generation Library" and "Generation Objects", you determine which objects will be put on the preload list:

- To put on the preload list all objects that are currently in the buffer pool, you enter an asterisk in each of the two fields (*).
- To put on the preload list certain of the objects in the buffer pool, you can use any combination of asterisks, full names and names with asterisk notation for both fields.

Example: To put on the preload list all objects which have been loaded from a specific library, you enter the full name of the library in the "Generation Library" field and an asterisk (*) in the "Generation Objects" field.

Objects from the library SYSBPMT will not be included in the generated preload list (as it can be assumed that these are objects which were only loaded into the buffer pool in order to execute this function).

> **Note:**
> All preload list objects will be generated as not resident (entry "N" in column "R") by default. Choose manually which objects you want to have resident.